Working Draft

Project T10/1611-D

Revision 04a
28 January 2008

Information technology - SCSI Stream Commands - 3 (SSC-3)

T10 Technical Editor:

David Peterson
Brocade Communications Inc.
6000 Nathan Lane North
Plymouth, MN 55442
USA

Telephone: 1-612-802-3299
Email: david.peterson@brocade.com

Reference number
ISO/IEC xxxxx-xxx : 200x
ANSI INCITS.*:200x**

Points of Contact:

T10 Chair
John B. Lohmeyer
LSI Logic
4420 Arrows West Drive
Colorado Springs, CO 80907-3444
Tel: 1-719-533-7560
Fax: 1-719-533-7183
Email: lohmeyer@t10.org

T10 Vice-Chair
George O. Penokie
IBM/Tivoli
MS 2C6
3605 Highway 52 North
Rochester, MN 55901
Tel: 1-507-253-5208
Fax: 1-507-253-2880
Email: gop@us.ibm.com

INCITS Secretariat
INCITS Secretariat
1250 Eye Street, NW Suite 200
Washington, DC 20005
Telephone: 1-202-737-8888
Facsimile: 1-202-638-4922
Email: incits@itic.org

T10 Web Site www.t10.org

T10 Reflector To subscribe send e-mail to majordomo@T10.org with 'subscribe' in message body
To unsubscribe send e-mail to majordomo@T10.org with 'unsubscribe' in message body
Internet address for distribution via T10 reflector: T10@T10.org

Document Distribution

INCITS Online Store
managed by Techstreet
1327 Jones Drive
Ann Arbor, MI 48105
http://www.techstreet.com/incits.html
Telephone: 1-734-302-7801
or 1-800-699-9277
Facsimile: 1-734-302-7811

or

Global Engineering
15 Inverness Way East
Englewood, CO 80112-5704
http://global.ihs.com
Telephone: 1-303-792-2181
or 1-800-854-7179
Facsimile: 1-303-792-2192

Rev 00 Changes:

- converted from SSC-2 rev 09
- changed references from SAM-2 to SAM-3
- 03-289r3: Only If Reserved Proposal
- removed asynchronous event reporting
- added SSC sense data descriptor from SPC-3 r17 (subclause 4.5.2.6 Stream commands sense data descriptor)

Rev 01 Changes:

- Completed hanging sentence in subclause 8.3.3 text for SWP bit and removed the redundent next paragraph.
- 04-123r0: Exclude REPORT LUNS from resetting TEST bit
- 04-119r0: DTD Status log page
- modified text for ASOCWP in 8.3.3
- modified text for PERSWP in 8.3.3
- modified text for PRMWP in 8.3.3

Rev 01a Changes:

- added text for SWP in 8.3.3

Rev 01b Changes:

04-211r1: WORM support for streaming devices
04-104r2: Add report to REPORT DENSITY SUPPORT command
04-178r2: Add device type specific VPD page
- broke READ POSITION command into subclauses

Rev 01c Changes:

04-283r1: Reservation model cleanup
05-014r0: Fix REPORT DENSITY SUPPORT data
05-037r0: Synchronize Mode Page Table with SPC-3
04-335r2: WORM Tamper Read Enable
05-373r2: Model subclause for WORM
05-066r0: Obsolete GAP SIZE field
05-067r0: Obsolete RSMK and setmarks

Rev 01d Changes:

05-002r4: Sequential-Access Device log page
05-068r2: Mode page support
05-159r1: Log page support

Rev 01e Changes:

05-334r0: Reinstate definition of Transfer Length field for WRITE FILEMARKS (note additional modifications were made to the draft standard to implement the change from TRANSFER LENGTH field to FILEMARK COUNT field)
05-320r1: Standardize processing of ERASE command with LONG=0
05-317r2: Remove Attached Media Changer model
05-262r3: ASC/ASCQ for Medium Thread Failure

Rev 02 Changes:

05-154r4: TapeAlert Enhancements

Rev 02a Changes:

- fixed error in Sequential-access device capabilities VPD length table (5-n changed to n)
05-213r5: Device Statistics log page for SSC-3 and Tape Diagnostic Data log page
05-155r4: Target Device Serial Number subpage
- changed SPC reference from SPC-3 to SPC-4

Rev 03 Changes:

06-172r1: Add commands to control data encryption
06-248r1: Add PREVENT ALLOW MEDIUM REMOVAL

Rev 03a Changes:

06-120r3: Secure Data Erase
06-180r1: Reporting Microcode Download in progress thru DT Activity and other fixes
06-293r1: Modification of the REPEAT bit behavior in the Tape Diagnostic Data log page
06-235r1: Align clean notification names
06-051r7: The Requirement for More than One Decryption Key

Rev 03b Changes:

05-049r6: Physical device model
06-385r1: Modifications to Tape Data Encryption protocol
06-391r0: Remove IV fields from the Data Encryption Algorithm descriptor

Rev 03c Changes:

- changed I_T nexus to I_T_L nexus in subclause 4.2.20.2 Encrypting data on the medium
05-140r1: Position after Self-Test
06-453r0: Add a random number page to the Tape Data Encryption protocol
06-412r3: Encryption KAD Lengths and Nonces
06-389r5: Using Public-Key Cryptography for Key wrapping

Rev 03d Changes:

07-005r2: Encryption unsupported medium
06-462r8: Keyless Copy of Encrypted Data
07-016r2: Additional controls for keyless copy
06-225r7: Using NIST AES Key-Wrap for Key Establishment
07-204r1: Fix conflict between 06-412r3 and 07-016r2

Rev 03e Changes:

07-218r3: Configurable Early Warning
07-254r1: Set Data Encryption Parameters through an SA

Rev 03f Changes:

06-138r6: TapeAlert Delineation

07-046r3: Requested Recovery log page
07-290r3: Protecting partially encrypted volumes

Rev 04 Changes:

07-361r6: Out of band encryption key management
08-022r1: Automation device serial number VPD page

Rev 04a (letter ballot version) Changes:

07-454r5: Capability-based Command Security

To Do:
- use the term logical block throughout the security text (see SCSIEdNote paragraph tag)
- review use of I_T nexus versus I_T_L nexus throughout the security text
- change encrypted data to encrypted logical block(s) and all associated field and bit names

ANSI (r)
INCITS.*:200x**

Draft

American National Standards for Information Systems -

SCSI Stream Commands - 3 (SSC-3)

Secretariat
National Committee for Information Technology Standards

Approved mm dd yy

American National Standards Institute, Inc.

Abstract

This standard specifies the device model and functional requirements for the SCSI sequential-access stream device type. This standards permits the SCSI sequential-access stream device type to attach to computers and provides the definitions for their use.

This standard does not contain material related to any service delivery subsystem which is used to transport the commands, command parameter data, command response data, and status specified in this standard.

Draft

American National Standard

Approval of an American National Standard requires verification by ANSI that the requirements for due process, consensus, and other criteria for approval have been met by the standards developer. Consensus is established when, in the judgment of the ANSI Board of Standards Review, substantial agreement has been reached by directly and materially affected interests. Substantial agreement means much more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered and that effort be made toward their resolution.

The use of American National Standards is completely voluntary; their existence does not in any respect preclude anyone, whether he or she has approved the standards or not, from manufacturing, marketing, purchasing, or using products, processes, or procedures not confirming to the standards.

The American National Standards Institute does not develop standards and will in no circumstances give interpretation on any American National Standard in the name of the American National Standards Institute. Requests for interpretations should be addressed to the secretariat or sponsor whose name appears on the title page of this standard.

CAUTION NOTICE: This American National Standard may be revised or withdrawn at any time. The procedures of the American National Standards Institute require that action be taken periodically to reaffirm, revise, or withdraw this standard. Purchasers of American National Standards may receive current information on all standards by calling or writing the American National Standards Institute.

CAUTION: The developers of this standard have requested that holders of patents that may be required for the implementation of the standard, disclose such patents to the publisher. However, neither the developers nor the publisher have undertaken a patent search in order to identify which, if any, patents may apply to this standard.

As of the date of publication of this standard and following calls for the identification of patents that may be required for the implementation of the standard, no such claims have been made. No further patent search is conducted by the developer or the publisher in respect to any standard it processes. No representation is made or implied that licenses are not required to avoid infringement in the use of this standard.

Published by
American National Standards Institute
11 West 42nd Street, New York, NY 10036

Printed in the United States of America

Contents

Tables

Figures

Foreword

This foreword is not part of American National Standard INCITS.***:200x.

This standard specifies the external behavior of a device server that defines itself as a sequential-access device in the DEVICE TYPE field of the INQUIRY command response data. This device type is known as a stream device. This standard conforms to the SCSI Architecture Model - 3 (T10/1561-D) standard.

With any technical document there may arise questions of interpretation as new products are implemented. INCITS has established procedures to issue technical opinions concerning the standards developed by INCITS. These procedures may result in SCSI Technical Information Bulletins being published by INCITS.

These Bulletins, while reflecting the opinion of the Technical Committee that developed the standard, are intended solely as supplementary information to other users of the standard. This standard, ANSI INCITS.***:200x, as approved through the publication and voting procedures of the American National Standards Institute, is not altered by these bulletins. Any subsequent revision to this standard may or may not reflect the contents of these Technical Information Bulletins.

Current INCITS practice is to make Technical Information Bulletins available through:

INCITS Online Store	http://www.techstreet.com/INCITS.html
managed by Techstreet	Telephone: 1-734-302-7801 or
1327 Jones Drive	1-800-699-9277
Ann Arbor, MI 48105	Facsimile: 1-734-302-7811

or

Global Engineering	http://global.ihs.com
15 Inverness Way East	Telephone: 303-792-2181 or
Englewood, CO 80112-5704	800-854-7179
	Facsimile: 303-792-2192

Requests for interpretation, suggestions for improvement and addenda, or defect reports are welcome. They should be sent to the INCITS Secretariat, InterNational Committee for Information Technology Standards, Information Technology Institute, 1250 Eye Street, NW, Suite 200, Washington, DC 20005-3922.

This standard was processed and approved for submittal to ANSI by the InterNational Committee for Information Technology Standards (INCITS). Committee approval of the standard does not necessarily imply that all committee members voted for approval. At the time of it approved this standard, INCITS had the following members:

<<Insert INCITS member list>>

The INCITS Technical Committee T10 on Lower Level Interfaces, which reviewed this standard, had the following members:

<<Insert T10 member list>>

Introduction

The SCSI Stream Commands - 3 (SSC-3) standard is divided into eight clauses:

Clause 1 is the scope.
Clause 2 enumerates the normative references that apply to this standard.
Clause 3 describes the definitions, symbols, and abbreviations used in this standard.
Clause 4 describes an overview and model of the sequential-access type device.
Clause 5 describes the explicit address command set for sequential-access type devices.
Clause 6 describes the implicit address command set for sequential-access type devices.
Clause 7 describes the common command set for sequential-access type devices.
Clause 8 describes the parameters for sequential-access type devices.

The annexes provide information to assist with implementation of this standard.

American National Standard **INCITS.***:200x**

American National Standard for Information Systems - Information Technology - SCSI Stream Commands - 3 (SSC-3)

1 Scope

This standard defines the command set extensions to facilitate operation of the sequential-access device type member of the SCSI stream device class. The clauses of this standard, implemented in conjunction with the applicable clauses of the SCSI Primary Commands - 3 standard, fully specify the standard command set for the sequential-access device type member of the SCSI stream device class.

The objectives of this standard are to provide the following:

a) permit an application client to communicate over a SCSI service delivery subsystem, with a logical unit that declares itself to be a sequential-access device in the device type field of the INQUIRY command response data;
b) define commands unique to the sequential-access device type; and
c) define commands to manage the operation of the sequential-access device type.

Figure 1 shows the relationship of this standard to the other standards and related projects in the SCSI family standards as of the publication of this standard.

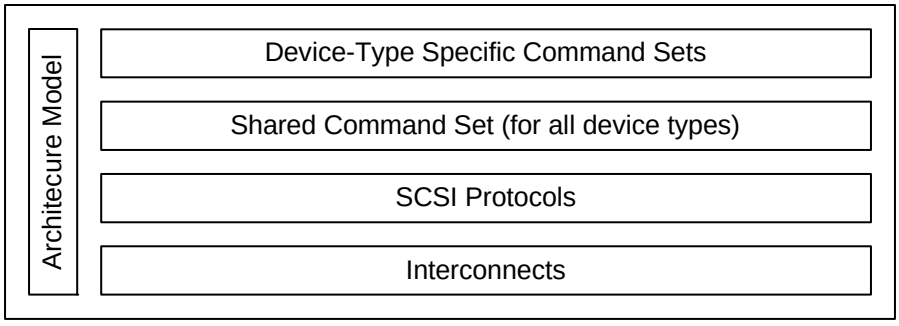

Figure 1 — SCSI document relationships

The roadmap in figure 1 is intended to show the general applicability of the documents to one another. The figure is not intended to imply a relationship such as a hierarchy, protocol stack, or system architecture. It specifies the applicability of a standard to the implementation of a given SCSI protocol.

This standard makes obsolete the following concepts from previous versions of this standard:

a) the GAP SIZE field;
b) RSMK and setmarks; and
c) Attached Media Changer model.

At the time this standard was generated, examples of the SCSI general structure included:

Physical Interconnects:		
Fibre Channel Arbitrated Loop	FC-AL	[ISO/IEC 14165-121] [ANSI X3.272:1996]
Fibre Channel Physical and Signalling Interface	FC-PH	[ISO/IEC 14165-111] [ANSI X3.230:1994]
Fibre Channel Physical Amendment 1		[ANSI X3.230/AM1:1996]
Fibre Channel 3rd Generation Physical Interface	FC-PH-3	[ISO/IEC 14165-113] [ANSI X3.303-1998]
Fibre Channel Framing and Signaling Interface	FC-FS	[T11/1331-D]
High Performance Serial Bus		[ANSI IEEE 1394:1995]
SCSI Parallel Interface - 2	SPI-2	[ISO/IEC 14776-112] [ANSI X3.302:1999]
SCSI Parallel Interface - 3	SPI-3	[ISO/IEC 14776-113] [ANSI NCITS.336:200x]
SCSI Parallel Interface - 4	SPI-4	[T10/1365-D]
Serial Storage Architecture Physical Layer 1	SSA-PH	[ANSI X3.293:1996]
Serial Storage Architecture Physical Layer 2	SSA-PH-2	[ANSI NCITS.307:1998]
Transport Protocols:		
Serial Storage Architecture Transport Layer 1	SSA-TL-1	[ANSI X3.295:1996]
Serial Storage Architecture Transport Layer 2	SSA-TL-2	[ANSI NCITS.308:1998]
SCSI-3 Fibre Channel Protocol	FCP	[ISO/IEC 14776-221] [ANSI X3.269:1996]
SCSI-3 Fibre Channel Protocol - 2	FCP-2	[ISO/IEC 14776-222] [T10/1144-D]
SCSI Fibre Channel Protocol - 3	FCP-3	[T10/1560-D]
SCSI Fibre Channel Protocol - 4	FCP-4	[T10/1828-D]
Serial Bus Protocol - 2	SBP-2	[ISO/IEC 14776-232] [ANSI NCITS.325:1999]
Serial Storage Architecture SCSI-3 Protocol	SSA-S3P	[ANSI NCITS.309:1998]
SCSI on Scheduled Transfer	SST	[T10/1380-D]
SCSI RDMA Protocol	SRP	[T10/1415-D]
Shared Command Sets:		
SCSI-3 Primary Commands	SPC	[ANSI X3.301:1997]
SCSI Primary Commands - 2	SPC-2	[ISO/IEC 14776-312] [T10/1236-D]
SCSI Primary Commands - 3	SPC-3	[ANSI/INCITS 408-2005] [T10/1416-D]
SCSI Primary Commands - 4	SPC-4	[T10/1731-D]
Device-Type Specific Command Sets:		
SCSI-3 Block Commands	SBC	[ISO/IEC 14776-321] [ANSI NCITS.306:1998]
SCSI Block Commands - 2	SBC-2	[T10/1417-D]
SCSI-3 Stream Commands	SSC	[ISO/IEC 14776-331] [ANSI NCITS.335:200x]
SCSI Stream Commands - 2	SSC-2	[T10/1434-D]
SCSI-3 Medium Changer Commands	SMC	[ISO/IEC 14776-351] [ANSI NCITS.314:1998]
SCSI Medium Changer Commands - 2	SMC-2	[T10/1383-D]
SCSI-3 Multimedia Command Set	MMC	[ANSI X3.304:1997]

SCSI Multimedia Command Set - 2	MMC-2	[ISO/IEC 14776-362] [ANSI NCITS.333:200x]
SCSI Multimedia Command Set - 3	MMC-3	[T10/1363-D]
SCSI-3 Controller Commands	SCC	[ISO/IEC 14776-341] [ANSI X3.276:1997]
SCSI Controller Commands - 2	SCC-2	[ISO/IEC 14776-342] [ANSI NCITS.318:1998]
SCSI Reduced Block Commands	RBC	[ANSI NCITS.330:200x]
SCSI Reduced MultiMedia Commands	RMC	[T10/1364-D]
SCSI-3 Enclosure Services Commands	SES	[ANSI NCITS.305:1998]
SCSI Specification for Optical Card Reader/Writer	OCRW	[ISO/IEC 14776-381]
Object-based Storage Devices Commands	OSD	[T10/1355-D]

Architecture Model:

SCSI-3 Architecture Model	SAM	[ISO/IEC 14776-411] [ANSI X3.270:1996]
SCSI Architecture Model - 2	SAM-2	[ISO/IEC 14776-412] [T10/1157-D]
SCSI Architecture Model - 3	SAM-3	[ANSI/INCITS 402-2005] [T10/1561-D]
SCSI Architecture Model - 4	SAM-4	[T10/1683-D]

The term SCSI is used to refer to the family of standards described in this clause.

2 Normative references

2.1 Normative references

The following standards contain provisions that, by reference in the text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the standards listed below.

Copies of the ISO/IEC documents may be obtained from ANSI: approved ANSI standards, approved and draft international and regional standards (ISO, IEC, CEN/CENELEC, ITUT), and approved and draft foreign standards (including BSI, JIS, and DIN). For further information, contact ANSI Customer Service Department at 212-642-4900 (phone), 212-302-1286 (fax) or via the World Wide Web at http://www.ansi.org.

Copies of the DoD document may be obtained via the World Wide Web at http://iase.disa.mil/policy.html#DoD.

2.2 Approved references

ISO/IEC 14776-411, *SCSI-3 Architecture Model standard*
ISO/IEC 14776-313, *SCSI Primary Commands - 3 standard*
ISO/IEC 14776-352, *SCSI Media Changer Commands - 2 standard*
DoD 5220.20-M, *National Industrial Security Program - Operating Manual*

2.3 References under development

At the time of publication, the following referenced standards were still under development. For information on the current status of the document, or regarding availability, contact the relevant standards body or other organization as indicated.

ISO/IEC 14776-xxx, *SCSI Architecture Model - 4 standard* [T10/1683-D]
ISO/IEC 14776-xxx, *SCSI Primary Commands - 4 standard* [T10/1731-D]

2.4 NIST references

Copies of the following approved NIST standards may be obtained through the National Institute of Standards and Technology (NIST) at http://csrc.nist.gov/publications/nistpubs/index.html.

NIST SP (Special Publication) 800-57 Recommendation for Key Management – Part 1: General, March 2007

3 Definitions, acronyms, keywords, and conventions

3.1 Definitions

3.1.1 additional sense code: See SPC-4.

3.1.2 application client: An object that is the source of SCSI commands and task management requests. Further definition of an application client may be found in SAM-3.

3.1.3 auxiliary memory: A memory component that is accessible to the device server. This memory is usually non-volatile and independent of the main function of the device server.

3.1.4 BOx: Either beginning-of-medium (see 3.1.5) or beginning-of-partition (see 3.1.6).

3.1.5 beginning-of-medium (BOM): The extreme position along the medium in the direction away from the supply reel that is accessible by the device. This position may not coincide with a beginning-of-partition position.

3.1.6 beginning-of-partition (BOP): The position at the beginning of the permissible recording region of a partition. This position may not coincide with a beginning-of-medium position.

3.1.7 block address mode: The mode of operation that the logical unit is currently supporting (see 4.2.16). The block address mode is either the explicit address mode (see 3.1.21) or the implicit address mode (see 3.1.29).

3.1.8 buffered mode: A mode of data transfer in write operations that facilitates tape streaming. Buffered mode is specified by a non-zero value (1h or 2h) in the BUFFER MODE field in the mode parameter header (see 8.3). Buffered mode is the opposite of unbuffered mode (see 3.1.79).

3.1.9 byte: An 8-bit construct.

3.1.10 command: A request describing a unit of work to be performed by a device server. A detailed definition of a command may be found in SAM-3.

3.1.11 command descriptor block (CDB): The structure used to communicate commands from an application client to a device server. A command descriptor block may have a fixed length of up to 16 bytes or a variable length of between 12 and 260 bytes.

3.1.12 common command: A command that is contained in both the explicit and implicit address command sets.

3.1.13 data encryption parameters: A set of parameters accessible through the Set Data Encryption page (see 8.5.3.2) that controls the data encryption and decryption process in the physical device (see 4.2.21.8).

3.1.14 device server: An object within a logical unit that processes SCSI tasks according to the rules of task management. A detailed definition of a device server may be found in SAM-3.

3.1.15 device type: The type of device (or device model) implemented by the device server.

3.1.16 early-warning (EW): A physical mark or device computed position near but logically before the end-of-partition, independent of physical direction (see 4.2.4).

3.1.17 encrypted block: A logical block containing data that has been subjected to a ciphering process by the device server. Within this standard, a logical block is only considered encrypted if the ciphering process was performed by a device server.

3.1.18 end-of-data (EOD): A recorded indication that no valid logical objects are recorded between this position and end-of-partition. End-of-data is denoted in a format-specific manner (see 4.2.7.1).

3.1.19 end-of-medium (EOM): The extreme position along the medium in the direction away from the take-up reel that is accessible by the device. This position may not coincide with a end-of-partition position.

3.1.20 end-of-partition (EOP): The position at the end of the permissible recording region of a partition.

3.1.21 explicit address mode: The mode of operation in which the logical unit is supporting the explicit address command set (see 3.1.22).

3.1.22 explicit address command set: The command set in which read and writes contain positioning information.

3.1.23 explicit command: A command contained only in the explicit address command set (see table 21).

3.1.24 field: A group of one or more contiguous bits, a part of a larger structure such as a CDB (see 3.1.11) or sense data (see 3.1.67).

3.1.25 filemark: A special recorded logical object within a partition, not containing user data, that provides a segmentation scheme for the contents of a partition.

3.1.26 fixed-block transfer: A read or write type command with the FIXED bit set to one.

3.1.27 format label: A vendor-specific series of logical objects that contain information used to identify the volume or data set.

3.1.28 generic command: An explicit command (see 3.1.23) that is not a read type or write type command.

3.1.29 implicit address mode: The mode of operation in which the logical unit is supporting the implicit address command set (see 3.1.30).

3.1.30 implicit address command set: The command set in which read and writes do not contain positioning information, and positioning is implied relative to the current position.

3.1.31 implicit command: A command contained only in the implicit address command set (see table 29).

3.1.32 INFORMATION field: A command-specific field in the sense data (see SPC-4).

3.1.33 I_T nexus: A nexus between a SCSI initiator port and a SCSI target port (see SAM-3).

3.1.34 I_T nexus loss: A condition resulting from the events defined by SAM-3 in which the SCSI device performs the operations described in SAM-3 and this standard.

3.1.35 I_T nexus loss event: A SCSI transport protocol specific event that triggers I_T nexus loss as described in SAM-3.

3.1.36 logical block: A logical object that is a unit of data supplied or requested by an application client.

3.1.37 logical file: Zero or more logical blocks starting immediately after BOP or a filemark (see 4.2.8).

3.1.38 logical file identifier: A unique identifier, within a partition, for a logical file (see 4.2.8.2).

3.1.39 logical identifier: A logical object identifier or logical file identifier.

3.1.40 logical object: A logical block or a filemark (see 4.2.7).

3.1.41 logical object identifier: A unique identifier, within a partition, for a logical object (see 4.2.7.2).

3.1.42 logical unit reset: A logical unit action in response to a logical unit reset event in which the logical unit performs the operations described in SAM-3.

3.1.43 logical unit reset event: An event that triggers a logical unit reset from a logical unit as described in SAM-3.

3.1.44 medium auxiliary memory (MAM): An auxiliary memory residing on a medium, for example, a tape cartridge.

3.1.45 message authentication code: Information used to validate the integrity of encrypted blocks (see 3.1.17).

3.1.46 native capacity: The capacity assuming one-to-one compression (e.g., compression disabled), the medium is in good condition, and that the device recommended typical block size is used.

3.1.47 nexus: A relationship between two SCSI devices, and the SCSI initiator port and SCSI target port objects within those devices (see SAM-3).

3.1.48 nonce: An unpredictable random value used only for a single instance or invocation of a cryptographic algorithm or protocol.

3.1.49 one: The logical true condition of a variable.

3.1.50 overlength: The incorrect-length condition that exists after processing a read command when the length of the actual logical block read exceeds the requested transfer length in the command descriptor block or the mode header block size field, whichever is appropriate.

3.1.51 page: Several commands use regular parameter structures that are referred to as pages. These pages are identified with a value known as a page code.

3.1.52 partition: The entire usable region for recording and reading in a volume or in a portion of a volume, defined in a vendor-specific or format-specific manner (see 4.2.6).

3.1.53 physical device: An object in a SCSI target device that performs operations on a volume (e.g., reading, writing, loading, and unloading).

3.1.54 principal density code: The principal density code is a density code selected by the device server. The logical unit specifies the principal density code by reporting a DEFLT bit of one in the density support data block descriptor for supported densities in response to the REPORT DENSITY SUPPORT command (see 7.8). The selection of the principal density code is vendor specific.

3.1.55 programmable-early-warning zone (PEWZ): A zone within a partition that has its EOP side established at early warning and extends towards BOP for a distance indicated by the PEWS field (see 8.3.8). See 4.2.5.

3.1.56 reservation loss: An event caused by the release of a reserve/release method reservation (see SPC-2) or by the transition within the device server from the state where a persistent reservation holder exists to the state where a persistent reservation holder does not exist (see SPC-4).

3.1.57 SCSI device: A device that contains one or more SCSI ports that are connected to a service delivery subsystem and supports a SCSI application protocol (see SAM-3).

3.1.58 SCSI domain: The interconnection of two or more SCSI devices and a service delivery subsystem. A detailed definition of a SCSI domain may be found in SAM-3.

3.1.59 SCSI initiator device: A SCSI device containing application clients and SCSI initiator ports that originates device service and task management requests to be process by a SCSI target device and receives device service and task management responses from the SCSI target devices. When used this term refers to SCSI initiator devices or SCSI target/initiator devices that are using the SCSI target/initiator port as a SCSI initiator port.

3.1.60 SCSI initiator port: A SCSI initiator device object that acts as the connection between application clients and the service delivery subsystem through which requests, indications, responses, and confirmations are routed. In all cases when this term is used it refers to a SCSI initiator port or a SCSI target/initiator port operating as a SCSI initiator port.

3.1.61 SCSI port: A SCSI device resident object that connects the application client, device server or task manager to the service delivery subsystem through which requests and responses are routed. SCSI port is synonmous with port. A SCSI port is one of: a SCSI initiator port, a SCSI target port, or a SCSI target/initiator port.

3.1.62 SCSI target device: A SCSI device containing logical units and SCSI target ports that receives device service and task management requests for processing and sends device service and task management responses to SCSI initiator devices. When used this term refers to SCSI target devices or SCSI target/initiator devices that are using the SCSI target/initiator port as a SCSI target port.

3.1.63 SCSI target port: A SCSI target device object that contains a task router and acts as the connection between device servers and task managers and the service delivery subsystem through which indications and responses are routed. When this term is used it refers to a SCSI target port or a SCSI target/initiator port operating as a SCSI target port.

3.1.64 SCSI target/initiator device: A SCSI device that has all the characteristics of a SCSI target device and a SCSI initiator device.

3.1.65 SCSI target/initiator port: A SCSI device resident object that has all the characteristics of a SCSI target port and a SCSI initiator port.

3.1.66 security meta-data: Data used by security methods to enable user data to be returned in the form it existed prior to the application of the security methods (e.g., data encryption parameters, passwords, wrapped keys). Security meta-data may be used for vendor-specific security methods.

3.1.67 sense data: See SPC-4.

3.1.68 sense key: See SPC-4.

3.1.69 service delivery subsystem: That part of an SCSI I/O system that transmits service requests to a logical unit or SCSI target device and returns logical unit or SCSI target device responses to a SCSI initiator device. A detailed definition of a service delivery subsystem may be found in SAM-3.

3.1.70 spacing: The act of positioning the medium on a sequential-access device while processing a SPACE command.

3.1.71 status: One byte of response information sent from a device server to an application client upon completion of each command. A detailed definition of status may be found in SAM-3.

3.1.72 synchronize operation: The process of writing buffered logical objects to the medium (see 4.2.10).

3.1.73 tagged write sequence: One or more WRITE(16), WRITE FILEMARKS(16), or ERASE(16) commands delineated by the FCS and LCS bits (see 5.6, 5.7, and 5.2).

3.1.74 tape: The medium on which data is recorded. The medium is normally a long thin medium that is spooled onto one or two reels, possibly within a cassette or cartridge.

3.1.75 TapeAlert: A device server cpapbility that provides detailed device diagnostic information using a standard interface.

3.1.76 thread: A part of the loading process in which the recording medium is being engaged for positioning on a suitable transport mechanism (e.g., spooled on to a take up reel, wrapped around the surface of a helical scan drum). After threading is complete the tape device may beginning positioning the medium to an initial position.

3.1.77 track: A contiguous line on the medium consisting of a pattern of recorded signals written by one write component.

3.1.78 track group: A set of tracks that are recorded at the same time.

3.1.79 unbuffered mode: The mode of operation where write data is written directly to the medium without being buffered. Unbuffered mode is specified by a zero value (0h) in the BUFFER MODE field in the mode parameter header (see 8.3). Unbuffered mode is the opposite of buffered mode (see 3.1.8).

3.1.80 underlength: The incorrect-length condition that exists after processing a read command when the requested transfer length in the command descriptor block or the mode header block size field, whichever is appropriate, exceeds the length of the actual logical block read.

3.1.81 unencrypted block: A logical block containing data that has not been subjected to a ciphering process by the device server.

3.1.82 unthread: A part of the unloading process in which the recording medium is being disengaged from the suitable transport mechanism (e.g., de-spooled from a take up reel, unwrapped from around the surface of a helical scan drum).

3.1.83 variable-block transfer: A read or write type command with the FIXED bit set to zero.

3.1.84 vendor-specific control meta-data: Vendor-specific information stored on the volume outside the user data area(s) that is used to control or specify how the volume is being used by application clients (e.g., directory information, partition information, EOD locations, copies of data stored in a vendor-specific manner, volume serial number information, number of logical blocks on the media).

3.1.85 volume: A recording medium together with its physical carrier.

3.1.86 zero: The logical false condition of a variable.

3.2 Acronyms

A-KAD	authenticated key-associated data
ALDC	Adaptive Lossless Data Compression: ISO/IEC 15200:1996
BOM	beginning-of-medium
BOP	beginning-of-partition
CbCS	Command-based command security
CDB	command descriptor block

DCLZ	Data Compression according to Lempel and Ziv: ISO/IEC 11558:1992
DEM	data encapsulation mechanism
ECC	error correction code
ECDSA	Elliptic Curve Digital Signature Algorithm
ECIES	Elliptic Curve Integrated Encryption Scheme
ECMA	European Computer Manufacturers Association
EOD	end-of-data
EOM	end-of-medium
EOP	end-of-partition
EW	early-warning
HC	hybrid cipher
IDRC	Improved Data Recording Capability
I/O	input-output
ID	identifier
KDF	key derivation function
KEM	key encapsulation mechanism
LSB	least significant bit
M	mandatory
M-KAD	metadata key-associated data
MA	MAC algorithm
MAC	message authentication code
MAM	medium auxiliary memory
MSB	most significant bit
NA	not applicable
O	optional
Rsvd	reserved
INCITS	InterNational Committee for Information Technology Standards
SA	security association
SAM-3	SCSI Architecture Model - 3
SBC	SCSI-3 Block Commands
SC	symmetric cipher
SCSI	Small Computer System Interface
SCSI-3	Small Computer System Interface - 3
SMC-2	SCSI Media Changer Commands - 2
SPC-2	SCSI Primary Commands - 2
SPC-4	SCSI Primary Commands - 4
SSC-3	SCSI Stream Commands - 3 (this standard)
U-KAD	unauthenticated key-associated data

3.3 Keywords

3.3.1 expected: A keyword used to describe the behavior of the hardware or software in the design models assumed by this standard. Other hardware and software design models may also be implemented.

3.3.2 invalid: A keyword used to describe an illegal or unsupported field or code value. Receipt of an invalid field or code value shall be reported as an error.

3.3.3 ignored: A keyword used to describe an unused field or code value. The contents or value of an ignored field or code value shall not be examined by the receiving SCSI device and may be set to any value by the transmitting SCSI device.

3.3.4 mandatory: A keyword indicating an item that is required to be implemented as defined in this standard.

3.3.5 may: A keyword that specifies flexibility of choice with no implied preference (equivalent to "may or may not").

3.3.6 may not: A keyword that specifies flexibility of choice with no implied preference (equivalent to "may or may not").

3.3.7 obsolete: A keyword indicating that an item was defined in prior SCSI standards but has been removed from this standard.

3.3.8 optional: A keyword that describes features that are not required to be implemented by this standard. However, if any optional feature defined by this standard is implemented, then it shall be implemented as defined in this standard.

3.3.9 reserved: A keyword referring to fields and code values that are set aside for future standardization. A reserved field shall be set to zero, or in accordance with a future extension to this standard. Recipients are not required to check reserved fields for zero values. Receipt of reserved code values in defined fields shall be reported as an error.

3.3.10 shall: A keyword indicating a mandatory requirement. Designers are required to implement all such mandatory requirements to ensure interoperability with other products that conform to this standard.

3.3.11 should: A keyword indicating flexibility of choice with a strongly preferred alternative; equivalent to the phrase "it is strongly recommended".

3.3.12 vendor specific: Items (e.g., fields, code values, etc.) that are not defined by this standard and may be vendor defined.

3.4 Editorial Conventions

Certain words and terms used in this standard have a specific meaning beyond the normal English meaning. These words and terms are defined either in clause 3.1 or in the text where they first appear. Names of commands, statuses, sense keys, additional sense codes, and additional sense code qualifiers are in all uppercase (e.g., REQUEST SENSE). Lowercase is used for words having the normal English meaning.

The names of fields are in small uppercase (e.g., ALLOCATION LENGTH). When a field name is a concatenation of acronyms, uppercase letter may be used for readability (e.g., NORMACA). Normal case is used when the contents of a field are being discussed. Fields containing only one bit are usually referred to as the name bit instead of the name field.

Numbers that are not immediately followed by lower-case b or h are decimal values.

Numbers immediately followed by lower-case b (xxb) are binary values.

Numbers or upper case letters immediately followed by lower-case h (xxh) are hexadecimal values.

The most significant bit of a binary quantity is shown on the left side and represents the highest algebraic value position in the quantity.

If a field is specified as not meaningful or the field is to be ignored, the entity that receives the field shall not take any action based on the value of that field.

Lists sequenced by letters (e.g., a-red, b-blue, c-green) show no priority relationship between the listed items. Numbered lists (e.g., 1-red, 2-blue, 3-green) show a priority ordering between the listed items.

If a conflict arises between text, tables, or figures, the order of precedence to resolve the conflicts is text; then tables; and finally figures. Not all tables or figures are fully described in the text. Tables show data format and values. Notes do not constitute any requirements for implementors.

The ISO convention of numbering is used (e.g., the thousands and higher multiples are separated by a space and a dot is used as the decimal point as in 65 536 or 0.5). Table 1 shows some examples of decimal numbers represented using the ISO and American conventions.

Table 1 — ISO and American numbering conventions examples

ISO	American
0.6	0.6
3.141 592 65	3.14159265
1 000	1 000
1 323 462.95	1 323 462.95

3.5 Notation Conventions

3.5.1 Notation for state diagrams

All state diagrams use the notation shown in figure 2.

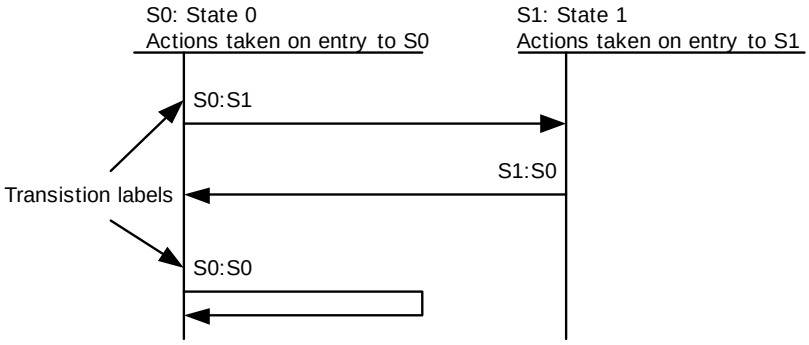

Figure 2 — Example state diagram

The state diagram is followed by a list of the state transitions, using the transition labels. Each transition is described in the list with particular attention to the conditions that cause the transition to occur and special conditions related to the transition. Using figure 2 as an example, the transition list might read as follows:

Transition S0:S1: This transition occurs when state S0 is exited and state S1 is entered.

Transition S1:S0: This transition occurs when state S1 is exited and state S0 is entered.

Transition S0:S0: This transition occurs when state S0 transitions to itself. It is particularly important to note that the actions taken whenever state S0 is entered are repeated every time this transition occurs.

A system specified in this manner has the following properties:

a) time elapses only within discrete states;
b) state transitions are logically instantaneous; and
c) every time a state is entered, the actions of that state are started. Note that this means that a transition that points back to the same state restarts the actions from the beginning.

4 General Concepts

4.1 Overview

The SCSI stream device class specifies the behavior of a logical unit that is primarily a streaming data device. Two device types are members of this class: sequential-access and printer devices. This standard addresses the sequential-access device type only. The sequential-access device type has the characteristic of primarily handling data in a sequential manner (i.e., a stream). This does not limit the device's ability to position randomly within the data although a sequential-access device is not truly random-access (see SBC-2 for a description of a random-access device).

This standard describes two modes and associated command sets for communicating with a sequential-access device:

a) implicit address mode. Commands to read and write on a sequential-access device do not contain any positioning information fields. Instead, the device position is normally determined by previous commands; and
b) explicit address mode. Commands to read and write on a sequential-access device contain positioning information fields.

Commands are available for absolute and relative positioning. Writing to a sequential-access device may cause all data starting at the point at which the data is written to be invalidated. There may be restrictions on where write operations may be initiated. Reading or writing data as a long string of data, as in a stream, tends to be the most efficient.

4.2 Sequential-access device model

4.2.1 Sequential-access device model overview

Sequential-access devices are described herein from the point of view of a tape device. However, other implementations are not precluded.

Sequential-access devices optimize their use in storing or retrieving user data in a sequential manner. Since access is sequential, position changes typically take a long time, when compared to random-access devices.

4.2.2 Physical elements

The recording medium for tape devices consists of various widths and lengths of a flexible substrate coated with a semi-permanent magnetic material. The recording medium may be spooled onto single reels or encapsulated into cartridges containing both a supply reel and a take-up reel. Several American National Standards exist covering the construction of reels and cartridges for interchange as well as recording techniques for many of the format or density combinations.

For a sequential-access device, a recording medium exists between two reels, the supply reel and take-up reel. The read/write mechanism may only access the medium between the reels. As the medium is taken out of one reel, it passes by the read/write mechanism and into the other reel. Transferring data as a stream is most efficient, since the medium may traverse the read/write mechanism producing a flow of data. To position to a given point requires moving the medium until the appropriate position is found.

The recording medium has two physical attributes called beginning-of-medium (BOM) and end-of-medium (EOM). Beginning-of-medium is at the end of the medium that is attached to the take-up reel. End-of-medium is at the end of the medium that is attached to the supply reel. In some cases, the medium is permanently affixed to one or both of the reel hubs. Beginning or end of medium is not required to be related to the beginning or end of any partition.

A volume is composed of the recording medium and its physical carrier (e.g., reel, cartridge, cassette). Volumes have an attribute of being mounted or demounted on a suitable transport mechanism.

Mounted is the state of a volume when the device is physically capable of processing commands that cause the medium to be moved. A volume is demounted when it is being loaded, threaded, unloaded, unthreaded, or when not attached to the device.

Ready is the state of the logical unit when medium access and non-medium access commands may be processed. The logical unit is not ready when no volume is mounted or, from the SCSI initiator device perspective, whenever any medium access command reports CHECK CONDITION status and a NOT READY sense key. The logical unit is not ready during the transition from mounted to not mounted, or not mounted to mounted. Devices may have a physical control that places the device in a not ready state even when a volume is mounted.

As shown in figure 3, a portion of the physical length of medium is not usable for recording data. For most volumes, a length of the medium is reserved between the take-up reel and the beginning-of-medium, and between the end-of-medium position and the supply reel. This is done to provide a sufficient tape wrap onto the reel hub and to ensure that recording starts in an undamaged section of the medium.

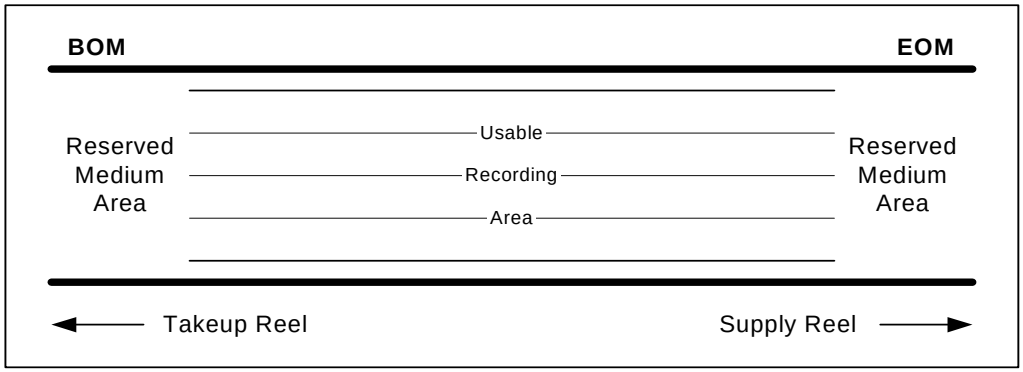

Figure 3 — Typical volume layout

The position on the medium where a pattern of recorded signals may be written by one write component is called a track (see figure 4). A device may write or read from one or more tracks at a time, depending on the format.

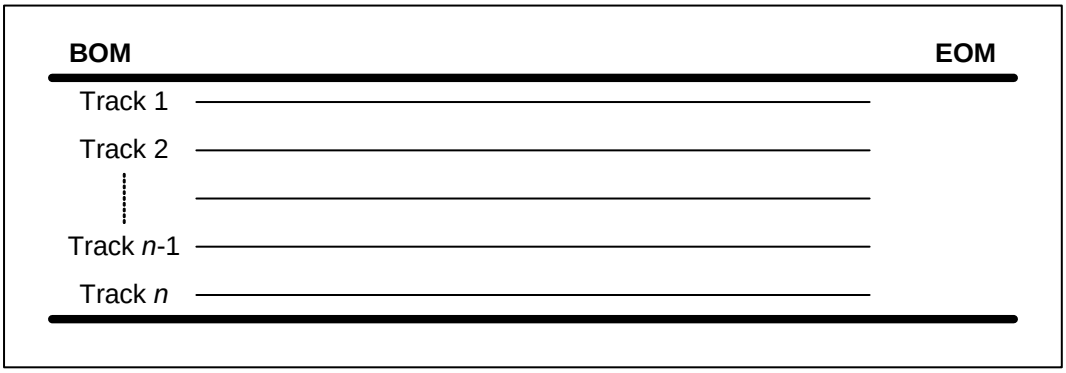

Figure 4 — Typical medium track layout

On a new volume, recording of one or more tracks begins after mounting the volume and moves from beginning-of-medium toward end-of-medium. The number of tracks written at one time is called a track group (TrkGrp). Track groups may be used by any recording format. For recorded volumes, reading in the forward direction follows the same course of tracks when writing.

In serpentine recording, not all tracks are recorded at the same time. At the end-of-medium or beginning-of-medium, the device reverses direction and begins recording the next track group. The process of reversing direction and recording the next track group may be repeated until all track groups are recorded. For serpentine devices that record only one track at a time, each physical track represents one track group (see figure 5).

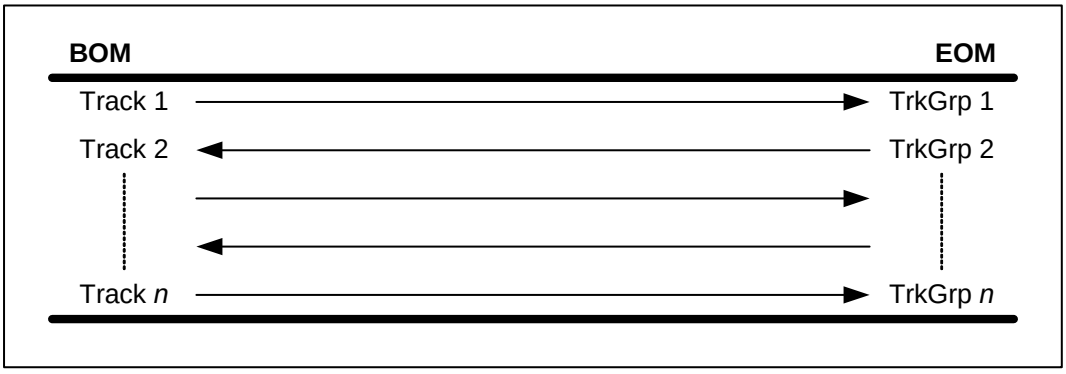

Figure 5 — Serpentine recording example

Some multi-track devices have only one track group, using a parallel storage format that supports the simultaneous recording of all available tracks (see figure 6).

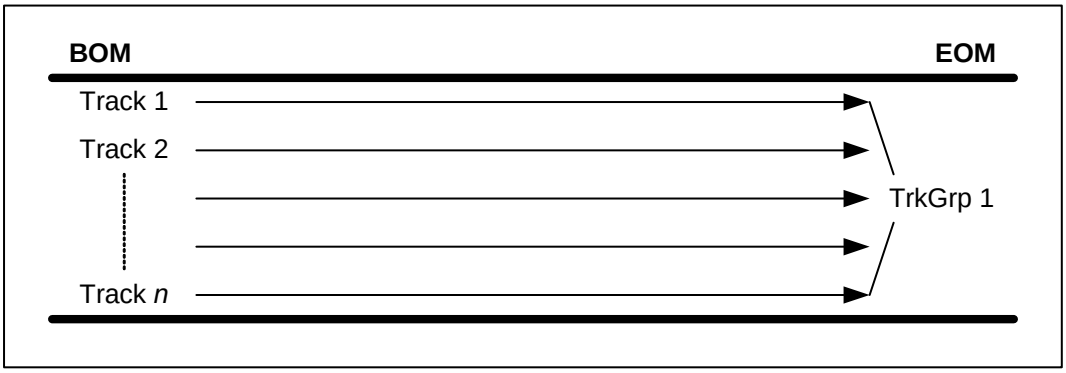

Figure 6 — Parallel recording example

The serpentine and parallel recording formats shown in the previous examples define tracks as longitudinal patterns of recorded information. One other storage format used by some devices records tracks diagonally across the medium. One or more tracks may be recorded at the same time. This recording technique is known as helical scan (see figure 7).

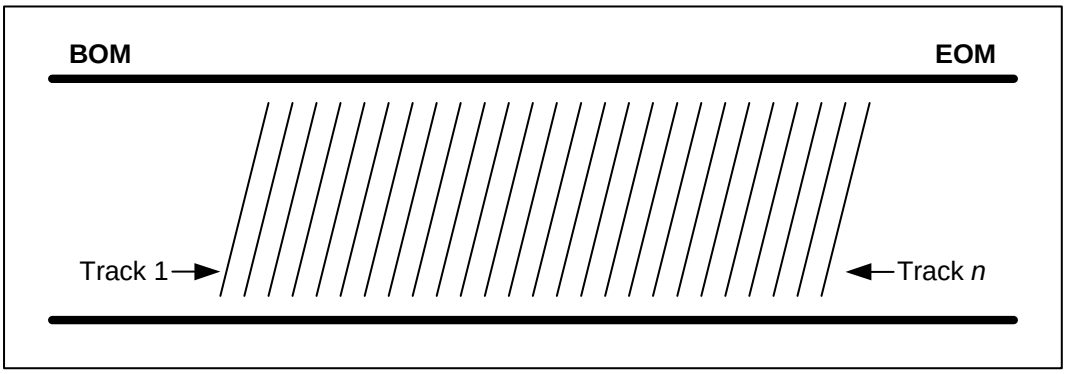

Figure 7 — Helical scan recording example

For most recording formats, a format identification in the form of a tone burst or some other recognizable pattern is recorded outside the user data area. The format identification is an attribute of a volume used for interchange purposes and is defined in applicable standards.

4.2.3 Physical device

A sequential-access device contains one or more physical devices. A physical device performs operations upon the medium (e.g., loading, unloading, positioning, writing and reading the medium, and reading and writing medium auxiliary memory).

The physical device is controlled by various entities, which may include:

a) one or more SCSI device servers (e.g., SSC and ADC);
b) an operator interface;
c) a management interface; and
d) a media changer.

A media changer may control the physical device by inserting a medium into or removing a medium from the physical device. When inserting a medium into or removing a medium from the physical device, the operator acts in the role of a media changer.

These entities perform operations that change various attributes of the physical device. These attributes affect the operations on a volume. Figure 8 shows in UML notation an example of the entities in a SCSI target device, and shows the attributes of the physical device.

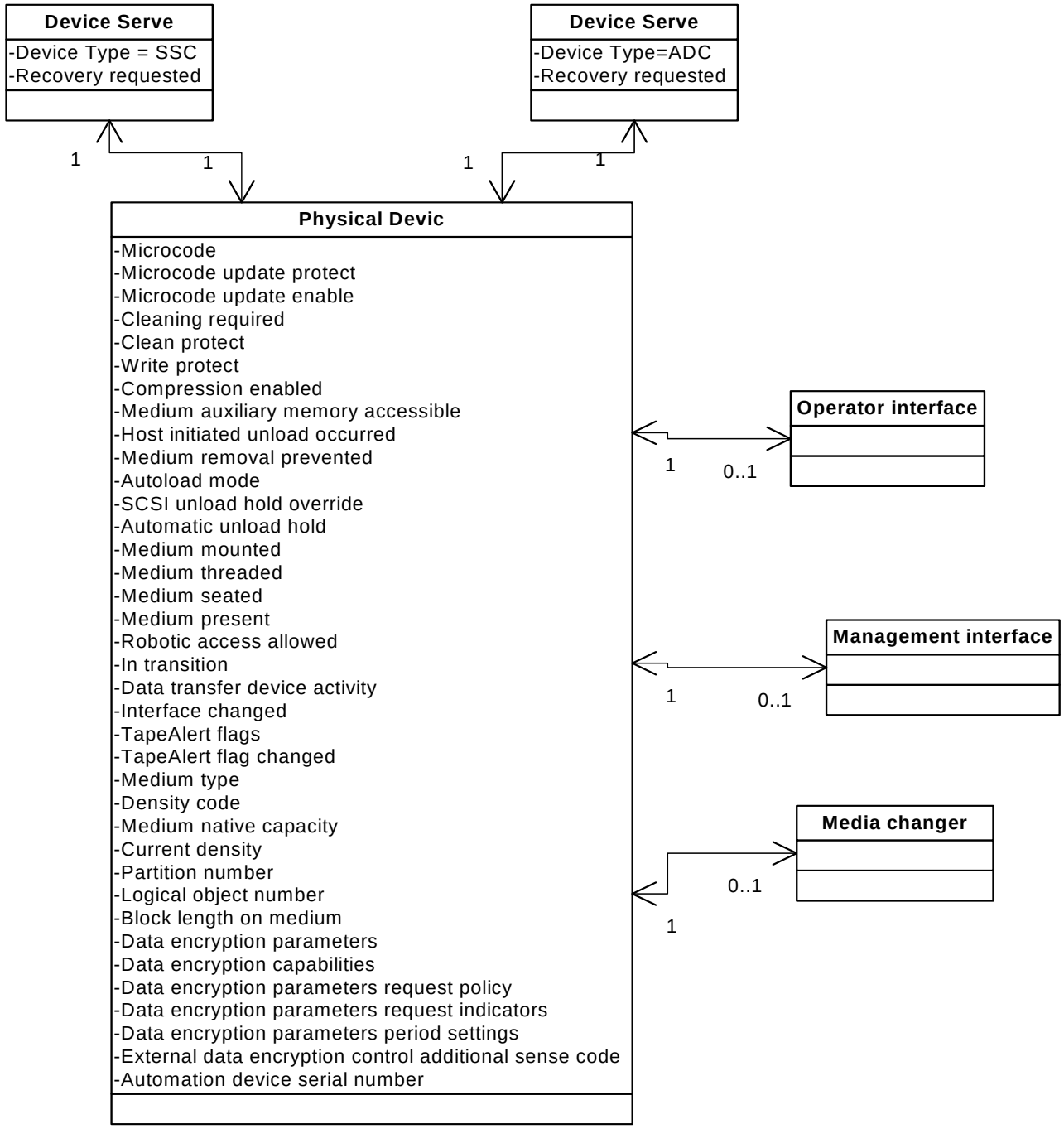

Figure 8 — UML example of SCSI target device and physical device

Table 2 specifies the standard that defines each attribute shown in figure 8..

Table 2 — Physical device attributes

Attribute	Reference
Microcode	SPC-4
Microcode update protect	ADC-2
Microcode update enable	ADC-2
Cleaning required	ADC-2
Clean protect	ADC-2
Write protect	ADC-2
Compression enabled	ADC-2
Medium auxiliary memory accessible	ADC-2
Host initiated unload occurred	ADC-2
Medium removal prevented	ADC-2
Autoload mode	SPC-4
SCSI unload hold override	ADC-2
Automatic unload hold	ADC-2
Medium mounted	ADC-2
Medium threaded	ADC-2
Medium seated	ADC-2
Medium present	ADC-2
Robotic access allowed	ADC-2
In transition	ADC-2
Data transfer device activity	ADC-2
Interface changed	ADC-2
TapeAlert flags	table 10
TapeAlert flag changed	ADC-2
Medium type	7.8.4
Density code	8.2.4.3
Medium native capacity[a]	7.8.3
Current density	ADC-2
Partition number	7.6.3
Logical object number	7.6.3
Block length on medium	SPC-4
Data encryption parameters	4.2.21.8

a) Medium native capacity is the value reported in the CAPACITY field of the density support data block descriptor when the MEDIA bit is one, and a SET CAPACITY command has not been used to affect the capacity of the medium.

Table 2 — Physical device attributes (Continued)

Attribute	Reference
Data encryption capabilities	4.2.21.9
Data encryption parameters request policy	4.2.22.3.2
Data encryption parameters request indicators	4.2.22.3.3
Data encryption parameters period settings	4.2.22.3.4
External data encryption control additional sense code	4.2.22.5
Automation device serial number	8.4.5
a) Medium native capacity is the value reported in the CAPACITY field of the density support data block descriptor when the MEDIA bit is one, and a SET CAPACITY command has not been used to affect the capacity of the medium.	

4.2.4 Early-warning

When writing, the application client needs an indication that it is approaching the end of the permissible recording area when moving in a direction toward the end of the partition (see 4.2.6). This position, called early-warning (EW), is typically reported to the application client at a position early enough for the device to write any buffered logical objects to the medium while still leaving enough room for additional recorded logical objects (see figure 9). Some American National Standards include physical requirements for a marker placed on the medium to be detected by the device as early-warning.

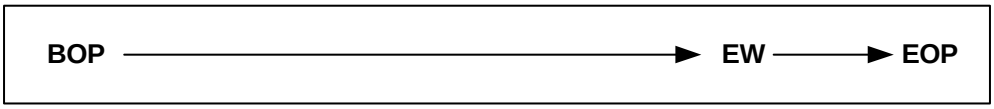

Figure 9 — Early-warning example

Devices are expected to report early warning to the application client when sufficient recording space is nominally available before EOP to record logical objects in the object buffer(s) and some additional logical objects. A logical concept of early-warning may be required to signal the application client at an appropriate location prior to the physical marker, particularly for devices that implement object buffers.

4.2.5 Programmable early warning

When writing, the application client may need an indication that it is approaching early warning (see 4.2.4) while there is enough space in the partition for the application client to write any buffered logical objects in the application client buffer to medium.

Application clients that need this indication may set the PEWS field (see 8.3.8) to a value that creates a PEWZ that allows sufficient recording space for the data that is in the application client buffer.

If the PEWZ is entered and exited before the PROGRAMMABLE EARLY WARNING DETECTED additional sense code is returned, the device server does not report PROGRAMMABLE EARLY WARNING DETECTED CHECK CONDITION.

At the completion of a WRITE(6), WRITE(16), WRITE FILEMARKS(6), WRITE FILEMARKS(16), ERASE(6), or ERASE(16) command that causes the medium to transition into the PEWZ the device server shall return CHECK CONDITION status, with the sense key set to NO SENSE, the EOM bit set to one and the additional sense code set to PROGRAMMABLE EARLY WARNING DETECTED. Encountering the PEWZ shall not cause the device server to perform a synchronize operation or terminate the command. If processing this command results in any other exception condition, the CHECK CONDITION status associated with that exception condition shall be reported instead. If the PROGRAMMABLE EARLY WARNING DETECTED additional sense was not reported, the next write in PEWZ that completes with GOOD status, shall return the programmable-early-warning CHECK CONDITION instead.

4.2.6 Partitions within a volume

Partitions consist of one or more non-overlapped logical volumes, each with its own beginning and ending points, contained within single physical volume. Each partition (*x*) within a volume has a defined beginning-of-partition (BOP *x*), an early-warning position (EW *x*), and an end-of-partition (EOP *x*).

All volumes have a minimum of one partition called partition 0, the default data partition. For devices that support only one partition, the beginning-of-partition zero (BOP 0) may be equivalent to the beginning-of-medium and the end-of-partition zero (EOP 0) may be equivalent to the end-of-medium. For devices that support more than one partition, they shall be numbered sequentially starting with zero (i.e., beginning-of-partition 0).

When a volume is mounted, it is logically positioned to the beginning of the default data partition (BOP 0). When a REWIND command is received in any partition (*x*), the device positions to the beginning-of-partition of the current partition (BOP *x*).

Partitions on a volume may be recorded in any order and use any partition number unique to the physical volume. It is sufficient for a device to be able to locate a partition, given its partition number, or to determine that it does or does not exist on the volume. For interchange, information about which partitions are present on a volume may be stored on the volume in a format specified area, possibly unavailable to the application client, or the information may be an intrinsic attribute of the device implementation.

Figure 10 shows a possible partition implementation for a four-track serpentine recording device, assuming that each track group defines a partition.

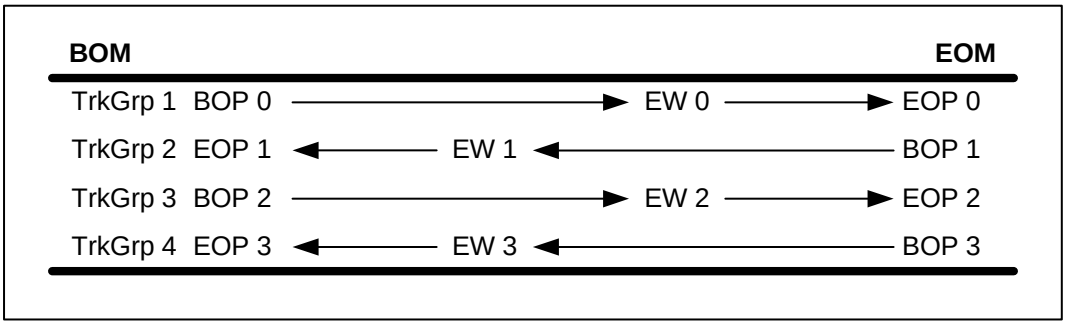

Figure 10 — Partitioning example - one partition per track group

Another possible partition implementation for this four-track serpentine recording device is shown in figure 11, using two track groups to define each partition.

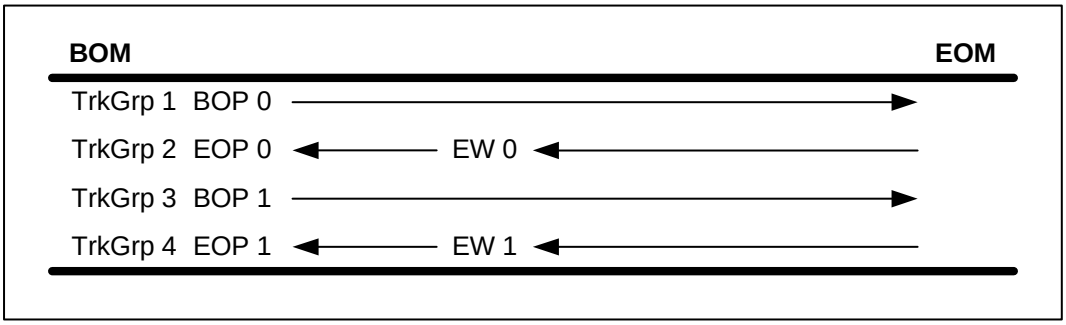

Figure 11 — Partitioning example - one partition per two track groups

The previous examples show the beginning and ending points for a partition aligned with physical bounds of the medium. This is not a mandatory requirement for partitioning. It is sufficient for a device to be able to locate to and stay in any partition bounded by a BOP *x* and EOP *x*. In this case, a device-recognizable attribute could be used to delineate the partitions. Figure 12 shows a possible two-partition implementation for a device with only one track group.

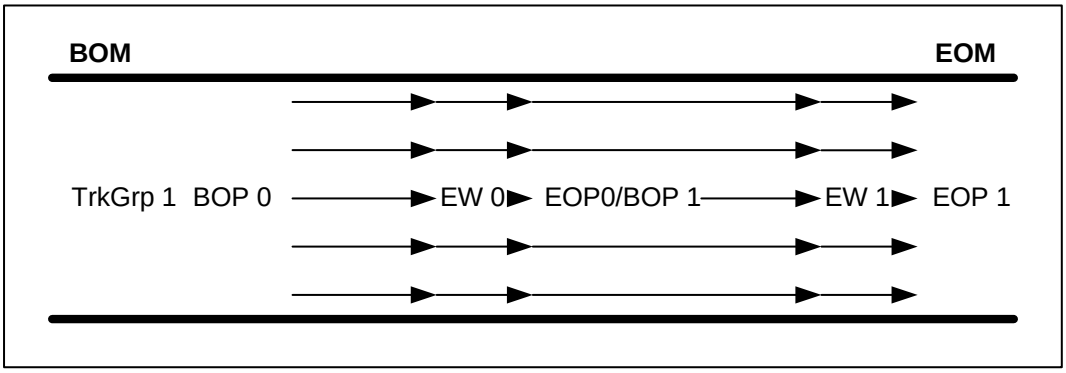

Figure 12 — Partitioning example - two partitions per track group

Three methods are defined in the MODE SENSE and MODE SELECT commands for managing partitions:

a) device-defined fixed locations;
b) device-defined based on an application client supplied number of partitions and a vendor-specific allocation algorithm; and
c) definition by partition number and capacity by an application client.

4.2.7 Logical objects

4.2.7.1 Logical objects within a partition

The area between BOP *x* and EOP *x* on a recorded volume contains application client accessible logical objects. Logical objects are controlled and transferred between the application client and the medium using commands defined in this standard. Each logical object shall have a logical object identifier that is unique within a partition.

The basic unit of data transferred by an application client is called a logical block. Logical blocks are stored according to the specifications of the format for the volume and may be recorded as portions of one or more physical blocks on the medium. The mapping between physical and logical blocks is the responsibility of the device server.

Filemarks are special recorded logical objects not containing user data. Proper recording and detection of filemarks is the responsibility of the device server. Application clients traditionally use filemarks to separate groups of user data from each other. Since some format standards do not define an explicit EOD, operating system software has often used conventions with filemarks to represent an EOD indication. In some implementations, the device's EOD definition may be specified by the application client using the Device Configuration mode page (see 8.3.3).

After writing data from BOP *x*, the medium is considered to be a contiguous grouping of logical objects. Depending on the format, blank medium may be treated as an end-of-data indication, an error recovery area, or an unrecoverable medium error causing an interchange error. Unrecorded volumes, new or erased, may exhibit blank medium characteristics if an attempt is made to read or space the volume before data has been written.

A sequential-access device may be capable of supporting fixed-block transfers or variable-block transfers. The concept of fixed or variable mode for writing and reading logical blocks only specifies the method by which the application client specifies the size of a logical block for transfer, and not the method of recording physical blocks on the medium. However, a device that supports only fixed-length physical blocks may only be capable of supporting logical blocks of the same length. The length of a logical block is always described in bytes. Refer to the READ BLOCK LIMITS command (see 7.5) for additional information about fixed-block transfers and variable-block transfers.

4.2.7.2 Logical object identifier

The logical object identifier value shall be a sequentially increasing number assigned to each logical object recorded in the partition starting with zero for the recorded logical object at BOP.

The READ POSITION command may be used to determine a logical object identifier and the application client may use this value with a LOCATE command or an explicit command to position to the same location at some future time.

4.2.8 Logical files

4.2.8.1 Logical files within a partition

Application clients may use filemarks to separate groups of user data into logical files. A logical file shall contain zero or more logical blocks and shall begin with the next logical object following a filemark or BOP. Each logical file shall have a logical file identifier that is unique within the partition.

4.2.8.2 Logical file identifier

The logical file identifier value shall be a sequentially increasing number assigned to each logical file recorded in the partition starting with zero for the recorded logical file beginning at BOP.

The READ POSITION command may be used to determine a logical file identifier and the application client may use this value with a LOCATE(16) command to position to the BOP side of the same logical file at some future time.

4.2.9 Object buffering

A device may contain a temporary storage area capable of holding one or more logical objects - an object buffer. A device object buffer may include any combination of logical objects in the process of being written to the medium, or it may contain read-ahead logical objects transferred from the medium.

A device with an object buffer may be capable of operating in either a buffered mode or an unbuffered mode. A device with no object buffer operates only in unbuffered mode. Either term is only applicable to the manner in which the device manages information to be written to the medium. Buffered mode is not applicable during read commands, regardless of whether read data passes through an object buffer.

A device operating in buffered mode may return GOOD status for write operations when all logical objects have been successfully transferred from the application client into the device object buffer. For devices operating in unbuffered mode, GOOD status is not returned until all requested logical objects are successfully recorded on the medium.

When issuing a buffered WRITE FILEMARKS command with the immediate bit set to one, GOOD status shall be returned as soon as the command is validated. For a WRITE FILEMARKS command with the immediate bit set to zero, the device server shall perform a synchronize operation (see 4.2.10).

If an unrecoverable write error occurs while in buffered mode, the device generates an error condition to the current active command. If no command is active, the error may be reported on the next applicable operation as a deferred error (see SPC-4). For some implementations auto contingent allegiance may be required. Refer to SAM-3 for a description auto contingent allegiance.

The READ POSITION command may be used to determine the number and storage space of buffered logical objects not written before the unrecoverable error was encountered.

A device that encounters an unrecoverable error during a read-ahead operation shall not report the error unless the logical object in error is accessed by an application client.

Prior to performing some commands, the device server shall perform a synchronize operation (see 4.2.10) as stated in table 21 and table 29.

4.2.10 Synchronize operation behavior

As stated in table 21 and table 29, some commands may require the device server to perform a synchronize operation (see 3.1.72). If a command requires a synchronize operation, the synchronize operation shall be performed prior to initiating any command-specific operations. Upon successful completion of the synchronize operation, no logical objects shall remain in the object buffer that have not been written to the medium. A synchronize operation shall have no effect on an object buffer that contains only read-ahead logical objects, or logical objects that have already been successfully written to the medium.

For a WRITE BUFFER command specifying modes 4, 5, 6, or 7 (download microcode operations), the device server shall perform a synchronize operation before performing the download operation.

For a MODE SELECT command specifying the Medium Partition mode page, the device server shall perform a synchronize operation before the logical unit partitions the medium.

For a SEND DIAGNOSTICS command, the device server shall perform a synchronization operation before any diagnostic tests that may affect the buffered logical objects, media, or logical position, are initiated.

4.2.11 Direction and position definitions

For sequential-access devices, positioning has the connotation of logically being in, at, before, or after some defined place within a volume. Positioning requires that the position is capable of being repeated under the same circumstances. The orientation of usage for the four words (in, at, before, or after) is in one direction, from BOP *x* toward EOP *x*. All positioning defined below is worded from this perspective. Devices without object buffers have some physical position that relates to these logical positions. However, these definitions do not require the medium to have a physical position equivalent to the logical position unless explicitly stated.

The forward direction is defined as logically progressing from BOP *x* toward EOP *x*. The reverse direction is defined as logically progressing from EOP *x* toward BOP *x*. In serpentine devices, the logical forward or reverse direction has an alternating relationship to the physical motion of the medium.

The concept of being in some position means not being outside a defined region. The definition allows the position to be on the boundary of a defined region. When a volume is first mounted, the logical position is always at the beginning of the default data partition (BOP 0). Whenever a volume is mounted and the medium motion is stopped, the position is in some partition. While moving between partitions, there is no stable position.

The concept of being at some position specifies being positioned to a logical or physical extremity of a partition. A sequential-access device may be positioned at BOM, at BOP *x*, at EOD, at EOP *x*, or EOM, since these are stable positions at extremities of a partition.

The concept of being before some position specifies that there is some logical object or other defined point that may be encountered when moving toward EOP *x*, if the proper commands are issued. Being positioned before a particular logical block means that if the device receives a valid READ command, the logical block is transferred to the application client. This position may also be before EW *x* and EOP *x*, since these are defined points within any partition. However, if data has not been written to the end-of-partition, these points may not be accessible by the SCSI initiator device.

The concept of being after some position specifies that there is some logical object or other defined point on the BOP *x* side of the current position that may be encountered if the proper commands are issued. When a READ command for a single logical block has been successfully processed, the logical position is after the transferred logical block.

4.2.12 Error reporting

4.2.12.1 Overview

Sequential-access devices compliant with this standard shall support both the fixed and descriptor sense data formats (see SPC-4). If fixed format sense data is specified, but the INFORMATION field value exceeds the maximum value allowed in the fixed format sense data, the VALID bit shall be set to zero. Refer to SPC-4 for a description of the sense data VALID bit and INFORMATION field contained in the REQUEST SENSE sense data. In addition, this standard describes the use of the INFORMATION field specific to the sequential-access device type.

4.2.12.2 Stream commands sense data descriptor

Table 3 defines the streams commands sense data descriptor for sequential-access devices.

Table 3 — Stream commands sense data descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE (04h)							
1	ADDITIONAL LENGTH (02h)							
2	Reserved							
3	FILEMARK	EOM	ILI	Reserved				

See SPC-4 for a description of the DESCRIPTOR TYPE and ADDITIONAL LENGTH fields.

See the READ(16) (see 5.3), READ(6) (see 6.4), SPACE(6) (see 6.6), RECOVER BUFFERED DATA (see 7.7), and SPACE(16) (see 7.11) commands for a description of FILEMARK bit usage.

See the READ(16) (see 5.3), READ REVERSE(16) (see 5.4), WRITE(16) (see 5.6), WRITE FILEMARKS(16) (see 5.7), LOCATE(10) (see 6.3), READ(6) (see 6.4), READ REVERSE(6) (see 6.5), SPACE(6) (see 6.6), WRITE(6) (see 6.8), WRITE FILEMARKS(6) (see 6.9), LOCATE(16) (see 7.3), RECOVER BUFFERED DATA (see 7.7), and SPACE(16) (see 7.11) commands, and the Device Configuration mode page (see 8.3.3) for end-of-medium (EOM) bit usage.

See the READ(16) (see 5.3) and READ(6) (see 6.4) commands and the Data Compression mode page (see 8.3.2) for incorrect length indicator (ILI) bit usage.

4.2.12.3 Information sense data descriptor

If reporting descriptor format sense data, the device server shall include an Information sense data descriptor (see table 4) for a unit attention condition generated:

a) by a TapeAlert specific informational exception condition; or
b) by a TapeAlert flag meeting its threshold criteria if the corresponding ETC bit in the TapeAlert log parameter is set to one.

The Information sense data descriptor format is specified in table 4.

Table 4 — Information sense data descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	DESCRIPTOR TYPE (00h)							
1	ADDITIONAL LENGTH (0Ah)							
2	VALID (1b)	Reserved						
3	FLAG01h	FLAG02h	FLAG03h	FLAG04h	FLAG05h	FLAG06h	FLAG07h	FLAG08h
4	FLAG09h	FLAG0Ah	FLAG0Bh	FLAG0Ch	FLAG0Dh	FLAG0Eh	FLAG0Fh	FLAG10h
5	FLAG11h	FLAG12h	FLAG13h	FLAG14h	FLAG15h	FLAG16h	FLAG17h	FLAG18h
6	FLAG19h	FLAG1Ah	FLAG1Bh	FLAG1Ch	FLAG1Dh	FLAG1Eh	FLAG1Fh	FLAG20h
7	FLAG21h	FLAG22h	FLAG23h	FLAG24h	FLAG25h	FLAG26h	FLAG27h	FLAG28h
8	FLAG29h	FLAG2Ah	FLAG2Bh	FLAG2Ch	FLAG2Dh	FLAG2Eh	FLAG2Fh	FLAG30h
9	FLAG31h	FLAG32h	FLAG33h	FLAG34h	FLAG35h	FLAG36h	FLAG37h	FLAG38h
10	FLAG39h	FLAG3Ah	FLAG3Bh	FLAG3Ch	FLAG3Dh	FLAG3Eh	FLAG3Fh	FLAG40h

The VALID bit shall be set to one.

An active TapeAlert flag has the corresponding FLAGXX bit set to one. An inactive TapeAlert flag has the corresponding FLAGXX bit set to zero.

4.2.12.4 Error conditions

If any of the following conditions occur during the processing of a command or if a deferred error prevented the command from processing, the device server shall return CHECK CONDITION status. The appropriate sense key and additional sense code should be set. Table 5 illustrates some error conditions and the applicable sense keys. Table 5 does not provide an exhaustive enumeration of all conditions that may cause the CHECK CONDITION status.

Table 5 — Error conditions and sense keys

Condition	Sense Key
Unsupported option requested.	ILLEGAL REQUEST
Logical unit reset, I_T nexus loss, or medium change since last command from this I_T nexus.	UNIT ATTENTION
Self diagnostic failed.	HARDWARE ERROR
Unrecovered read error.	MEDIUM ERROR HARDWARE ERROR
Recovered read or write error.	RECOVERED ERROR
Attempt a WRITE, READ, READ REVERSE, VERIFY, or RECOVER BUFFERED DATA command with the FIXED bit set to zero and variable-block transfers are not supported.	ILLEGAL REQUEST

Table 5 — Error conditions and sense keys (Continued)

Condition	Sense Key
Attempt a WRITE, READ, READ REVERSE, VERIFY, or RECOVER BUFFERED DATA command with the FIXED bit set to zero and requested block length is not supported.	ILLEGAL REQUEST
Attempt a WRITE, READ, READ REVERSE, VERIFY, or RECOVER BUFFERED DATA command with the FIXED bit set to one and MODE SENSE block length set to zero.	ILLEGAL REQUEST
Attempt to perform an erase, format, partition, set capacity, or write-type operation on write protected medium.	DATA PROTECT
Deferred write error.	MEDIUM ERROR VOLUME OVERFLOW HARDWARE ERROR
Medium failed to thread or unthread during the process of mounting or demounting.	MEDIUM ERROR HARDWARE ERROR

See the READ(16) (see 5.3), READ REVERSE(16) (see 5.4), VERIFY(16) (see 5.5), WRITE(16) (see 5.6), READ(6) (see 6.4), READ REVERSE(6) (see 6.5), VERIFY(6) (see 6.7), WRITE(6) (see 6.8), and RECOVER BUFFERED DATA (see 7.7) commands for a description of the FIXED bit.

The Read-Write Error Recovery mode page (see 8.3.5) current values specify behavior when an unrecoverable read or write error is encountered. If the Read-Write Error Recovery mode page is not implemented, the behavior is vendor specific.

In the case of a deferred write error, the sense data VALID bit shall be set to zero.

In the case of an unrecovered write error or a deferred write error, if buffered mode 1h is selected, the error shall be reported to the first application client issuing a command (other than INQUIRY or REQUEST SENSE). If buffered mode 2h is selected, the error shall be reported to the SCSI initiator device with unwritten data in the object buffer.

In the case of a write attempt to write protected medium, the additional sense code specifies the cause of the DATA PROTECT sense key (see 4.2.13).

In the case of a medium thread or unthread failure, the additional sense code shall be set to MEDIUM THREAD OR UNTHREAD FAILURE. The sense key shall be set to MEDIUM ERROR or HARDWARE ERROR (see SPC-4).

4.2.13 Write protection

4.2.13.1 Write protection introduction

Write protection of the medium prevents the alteration of logical objects on the medium and any change to the accessibility of logical objects on the medium, by commands issued to the device server. Write protection is usually controlled by the user of the medium through manual intervention (e.g., mechanical lock) or may result from hardware controls (such as tabs on the media housing), conditions such as positioning within unrecoverable data, or software write protections. All sources of write protection are independent. When present, any write protection shall cause otherwise valid commands that request alteration of logical objects on the medium, or affect the accessibility of logical objects on the medium, to be rejected with a CHECK CONDITION status with the sense key set to DATA PROTECT (see 4.2.13.2). Only when all write protections are disabled shall the device server process commands that request alteration of logical objects on the medium, or commands that may affect the accessibility of logical objects on the medium.

Hardware write protection results when a physical attribute of the drive or medium is changed to specify that writing shall be prohibited. Changing the state of the hardware write protection requires physical intervention, either with the drive or the medium. If allowed by the drive, changing the hardware write protection while the medium is mounted results in vendor-specific behavior that may include the writing of previously buffered logical objects.

Conditions such as positioning within unrecoverable data may result in a temporary write protection condition. To preserve future data integrity, the device server may reject any command that requires writing data to the medium when the recovery of the data is uncertain. A temporary write protection condition may be released by the device server at any time. Buffered logical objects may or may not be written to the media (e.g., the application client unloads the volume before the temporary write protection condition is removed). The exact behavior of the device server during a temporary write protection condition is vendor specific.

Software write protection results when either the device server or medium is marked as write protected by a command from the application client. Four optional means of setting a software write protection state are available to an application client through the Device Configuration and Control mode pages:

a) software write protection for the device server across mounts;
b) associated write protection for the currently mounted volume;
c) persistent write protection of a volume across mounts; and
d) permanent write protection of a volume across mounts.

The application client may control these write protections using the MODE SELECT command with the Control mode page (see SPC-4) and the Device Configuration mode page (see 8.3.3). All of the software write protection methods are optional. Changing the state of any software write protection shall not prevent previously buffered logical objects from transferring to the media.

4.2.13.2 Write protection additional sense code use

The additional sense code associated with the DATA PROTECT sense key depends on the write protection in effect at the time. Table 6 specifies the preferred additional sense code for the given write protection. Alternatively, the generic additional sense code of WRITE PROTECTED may be returned by the device server.

Table 6 — Write protect additional sense code combinations

Cause of DATA PROTECT error	Additional Sense Code
Hardware Write Protection	HARDWARE WRITE PROTECTED
Permanent Write Protection	PERMANENT WRITE PROTECT
Persistent Write Protection	PERSISTENT WRITE PROTECT
Associated Write Protection	ASSOCIATED WRITE PROTECT
Software Write Protection	LOGICAL UNIT SOFTWARE WRITE PROTECTED

If more than one condition exists, the device server shall either report the applicable condition in order of HARDWARE WRITE PROTECTED, PERMANENT WRITE PROTECT, PERSISTENT WRITE PROTECT, ASSOCIATED WRITE PROTECT, and LOGICAL UNIT SOFTWARE WRITE PROTECTED, or report the generic additional sense code of WRITE PROTECTED.

Other conditions that may cause a command to be rejected with a DATA PROTECT sense key include:

a) the format on the current medium is read-only by the device server;
b) the device server can only write from BOP or EOD and the current logical position is neither;
c) the medium is an archive tape and only can be recorded at EOD; and
d) vendor-specific conditions.

4.2.13.3 Software write protection for the device server

Software write protection for the device server controls write protection for the device server. This method of write protection is optionally controlled from the Control mode page (see SPC-4) or the SWP bit in the Device Configuration mode page (see 8.3.3). Either or both methods may be implemented by the device server. If both methods are implemented, each control bit is independently set. Software write protection exists if either bit is non-zero. The state of software write protection for the device server shall not be recorded on media. The value of the SWP bit may be altered by the application client (if the SWP bit is changeable). The state of each control bit shall be set to its default state after a logical unit reset.

4.2.13.4 Associated write protection

Associated write protection controls write protection for the currently mounted volume as long as the current volume is mounted. The associated write protection state is controlled by the ASOCWP bit in the Device Configuration mode page (see 8.3.3). Associated write protection exists if the ASOCWP bit is non-zero. Associated write protection may be altered by the application client (if the ASOCWP bit is changeable) if a volume is mounted. If a volume is demounted or after a logical unit reset occurs, associated write protection shall be removed.

4.2.13.5 Persistent write protection

Persistent write protection controls write protection for the currently mounted volume. The persistent write protection state is controlled by the PERSWP bit in the Device Configuration mode page (see 8.3.3). If enabled, persistent write protection shall exist for the mounted volume until disabled by the application client. The state of persistent write protection shall be recorded with the volume and the persistent write protection shall only affect the application client accessible medium. The device server shall report the PERSWP bit as one when a mounted volume is marked with persistent write protection. If a volume is demounted or after a logical unit reset occurs, the device server shall report the PERSWP bit as zero prior to the mounting of a volume. The means for recording the state of persistent write protection for the volume may be specified in the applicable recording format standard or be vendor specific.

4.2.13.6 Permanent write protection

Permanent write protection controls write protection for the currently mounted volume. The permanent write protection state is controlled by the PRMWP bit in the Device Configuration mode page (see 8.3.3). If enabled, permanent write protection shall exist for the mounted volume until disabled by a vendor-specific method. The state of permanent write protection shall be recorded with the volume and the persistent write protection shall only affect the application client accessible medium. The device server shall report the PRMWP bit as one when a mounted volume is marked with permanent write protection. If a volume is demounted or after a logical unit reset occurs, the device server shall report the PRMWP bit as zero prior to the mounting of a volume. The means for recording the state of permanent write protection for the volume may be specified in the applicable recording format standard or be vendor specific. Permanent write protection shall not be removed by a MODE SELECT command using the PRMWP bit. Methods to remove this protection may or may not exist and are vendor specific.

4.2.14 Progress indication

For the following immediate operations where the device server remains ready, an application client may test the progress of the operation (see table 7).

Table 7 — Commands providing progress indication without changing ready state

Operation	Options	Subclause	Additional Sense Code
ERASE	LONG = 1	5.2,6.2	ERASE OPERATION IN PROGRESS
LOCATE		5.2,6.3	LOCATE OPERATION IN PROGRESS

Table 7 — Commands providing progress indication without changing ready state

Operation	Options	Subclause	Additional Sense Code
REWIND		7.9	REWIND OPERATION IN PROGRESS
SET CAPACITY		7.10	SET CAPACITY OPERATION IN PROGRESS
VERIFY		5.5,6.7	VERIFY OPERATION IN PROGRESS

While the device server is performing the operation, an application client may test the progress of the operation by interpreting the progress indication information in the sense-key specific field of the sense data. During the operation, the device server may report a sense key value of NO SENSE and additional sense code as specified in table 7. The device server should use the sense key specific function for progress indication to provide information on the completion of the operation.

For the following immediate operations where the device server is ready or will become ready, an application client may follow the progress of the operation (see table 8).

Table 8 — Commands changing ready state and providing progress indication

Operation	Options	Subclause	Additional Sense Code
FORMAT MEDIUM		7.1	LOGICAL UNIT NOT READY, FORMAT IN PROGRESS
LOAD UNLOAD	LOAD = 1, EOT = 0	7.2	LOGICAL UNIT IS IN PROCESS OF BECOMING READY
LOAD UNLOAD	LOAD = 0, EOT = 1	7.2	LOGICAL UNIT NOT READY, OPERATION IN PROG-RESS

While the device server is performing the operation, an application client may test the progress of the operation by interpreting the progress indication information in the sense-key specific field of the sense data. During the operation, the device server may report a sense key value of NOT READY and an additional sense code as specified in table 8. The sense key specific function for progress indication may be used by the device server to provide information on the completion of the operation.

NOTE 1 A REQUEST SENSE command following a TEST UNIT READY command that results in CHECK CONDITION status may provide information that, if acted upon, may lead to unexpected conditions. For example, progress indication reporting is useful when a media changer is used to service a sequential-access device following an unload operation with IMMED=1b. A TEST UNIT READY command may receive CHECK CONDITION status and a NOT READY sense key reported in the subsequent sense data. This may imply that the unload operation is finished. If the application client ignores the progress indication information in the sense data, an EXCHANGE MEDIUM or MOVE MEDIUM command (see SMC-2) to move the demounted volume from the SCSI device may fail to grab the volume if the unload operation is still in progress.

4.2.15 Tagged command queuing

4.2.15.1 Tagged command queuing overview

A device server may choose to implement support for tagged tasks (i.e., command queuing). Issuing tagged write commands with object buffering disabled can facilitate streaming operations up to the limit of the number of outstanding tagged commands supported by the application client and the device server. This limit may effectively reduce the usable portion of the object buffer which may affect device server performance.

NOTE 2 For proper operation when performing tagged command queuing operations, an application client should wait for status to be returned for any MODE SELECT command before issuing the next command.

NOTE 3 Since the EXTENDED COPY command does not specify an immediate bit, the use of tagged command queuing may be required to prevent an overlapped command condition (e.g., when querying the progress of the extended copy operation using the RECEIVE COPY RESULTS command).

4.2.15.2 Explicit address mode tagged write sequences

When operating in explicit address mode, tagged write sequences (see 3.1.73) are used to support tagged command queuing for write operations.

For explicit address mode tagged write sequences, the following rules apply:

a) for a tagged write sequence consisting of more than one command, the FCS bit (see 5.2, 5.6, 5.7) shall be set to one in the first command of the tagged write sequence. For all other commands within the tagged write sequence, the FCS bit shall be set to zero;
b) for a tagged write sequence consisting of more than one command, the LCS bit (see 5.2, 5.6, 5.7) shall be set to one in the last command of the tagged write sequence. For all other commands within the tagged write sequence, the LCS bit shall be set to zero;
c) for a tagged write sequence consisting of only one command, the FCS bit and LCS bit shall be set to one;
d) for a tagged write sequence consisting of more than one command, the application client shall issue the commands in sequentially-increasing logical object identifier order;
e) an application client shall not issue a tagged write sequence prior to receiving status for all outstanding read type commands; and
f) an application client shall specify a Command Reference Number (see SAM-3) for each command in a tagged write sequence.

4.2.16 Block address mode

4.2.16.1 Block address mode overview

When operating in implicit address mode, spacing operations and commands to read and write on the medium do not contain positioning information (i.e., a command is processed based on the medium position relative to the last command that was processed). As such, the application client does not contain explicit knowledge of the medium position. If an error occurs, to maintain data integrity, the application client must determine the medium position before re-issuing a command that affects the medium position.

When operating in explicit address mode, commands to read and write on the medium contain positioning information fields (i.e., a command is processed based on the medium position information specified in the command). As such, the application client contains explicit knowledge of the medium position. This results in enhanced error detection and recovery functionality that allows the application client to maintain data integrity while performing various operations without first determining the medium position. Some example operations include:

a) the re-issuing of a command that affects the medium position;
b) multi-path I/O; and
c) tagged command queuing.

The device server shall support explicit address mode only, implicit address mode only, or both explicit address mode and implicit address mode. At any instance, the device server shall be in or transitioning between one of the following block address mode states (see 4.2.16.3):

a) A0:Idle;
b) E0:Explicit Address Mode - Neutral;
c) E1:Explicit Address Mode - Write Capable; or
d) F0:Implicit Address Mode.

4.2.16.2 Block address mode selection

The block address mode shall be selected as follows:

a) if the BAML bit in the Device Configuration mode page (see 8.3.3) is set to zero, then the setting of the BAM bit in the Device Configuration mode page (see 8.3.3) shall be ignored and the block address mode shall be determined based on the first block address mode unique command that is received after a successful load operation or successful completion of a command that positions the medium to BOP;
b) if the BAML bit in the Device Configuration mode page is set to one and the BAM bit in the Device Configuration mode page is set to zero, the logical unit shall support implicit address mode; or
c) if the BAML bit in the Device Configuration mode page is set to one and the BAM bit in the Device Configuration mode page is set to one, the logical unit shall support explicit address mode.

Prior to performing a block address mode change, the logical unit shall perform a synchronize operation (see 4.2.10).

If the device server receives a command that is not supported by the currently selected mode, the device server shall return CHECK CONDITION. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to:

a) ILLEGAL COMMAND WHILE IN EXPLICIT ADDRESS MODE if the currently selected mode is the explicit address mode;
b) ILLEGAL COMMAND WHILE IN IMPLICIT ADDRESS MODE if the currently selected mode is the implicit address mode; or
c) ILLEGAL COMMAND WHILE IN WRITE CAPABLE STATE if the device server is in explicit address mode write capable state.

4.2.16.3 Block address mode state diagrams

For the block address mode state diagrams (see figure 14, figure 15, figure 16, and figure 17), the following terminology shall apply:

a) explicit command: a command contained only in the explicit address command set (see table 21);
b) implicit command: a command contained only in the implicit address command set (see table 29); and
c) generic command: an explicit command that is not a read type or write type command (see table 21).

A common command containing a BAM bit (e.g., LOCATE(16)) shall be processed as either an explicit or implicit command based on the setting of the BAM bit contained in the common command.

The SPACE(16) command shall be processed as either an explicit or implicit command based on the setting of the PARAMETER LENGTH field in the command descriptor block.

Figure 13 provides an overview of the block address model state diagram. Refer to figure 14, figure 15, figure 16, and figure 17 for detailed descriptions of the block address model state diagram.

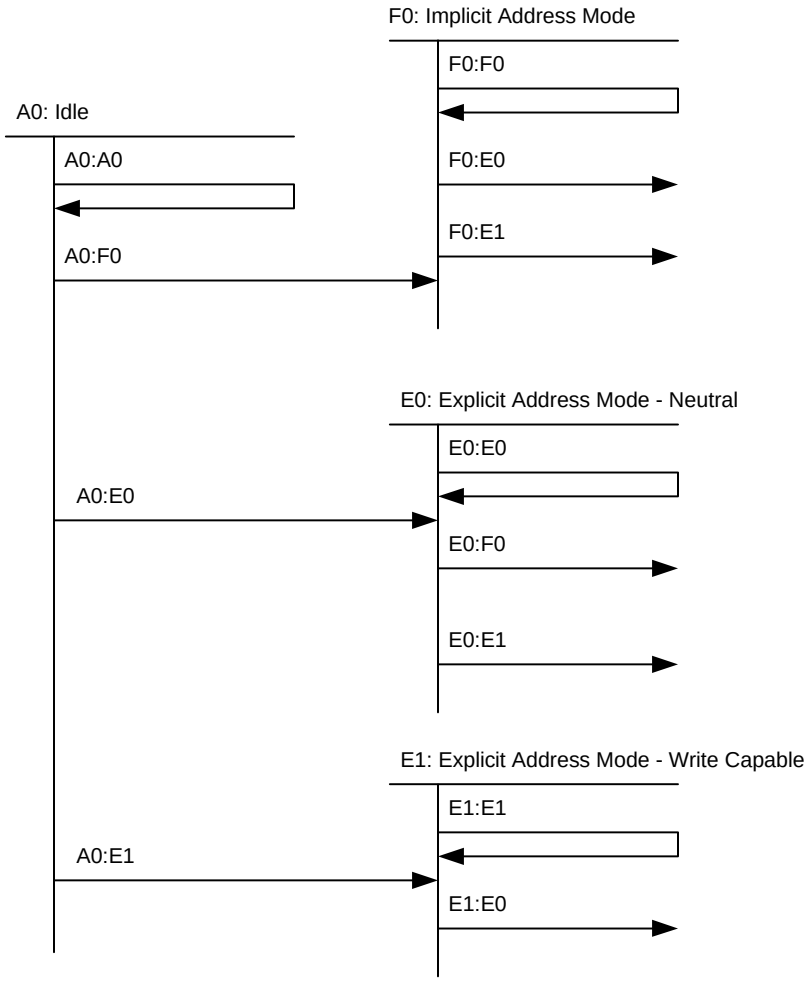

Figure 13 — Block address mode state diagram, overview

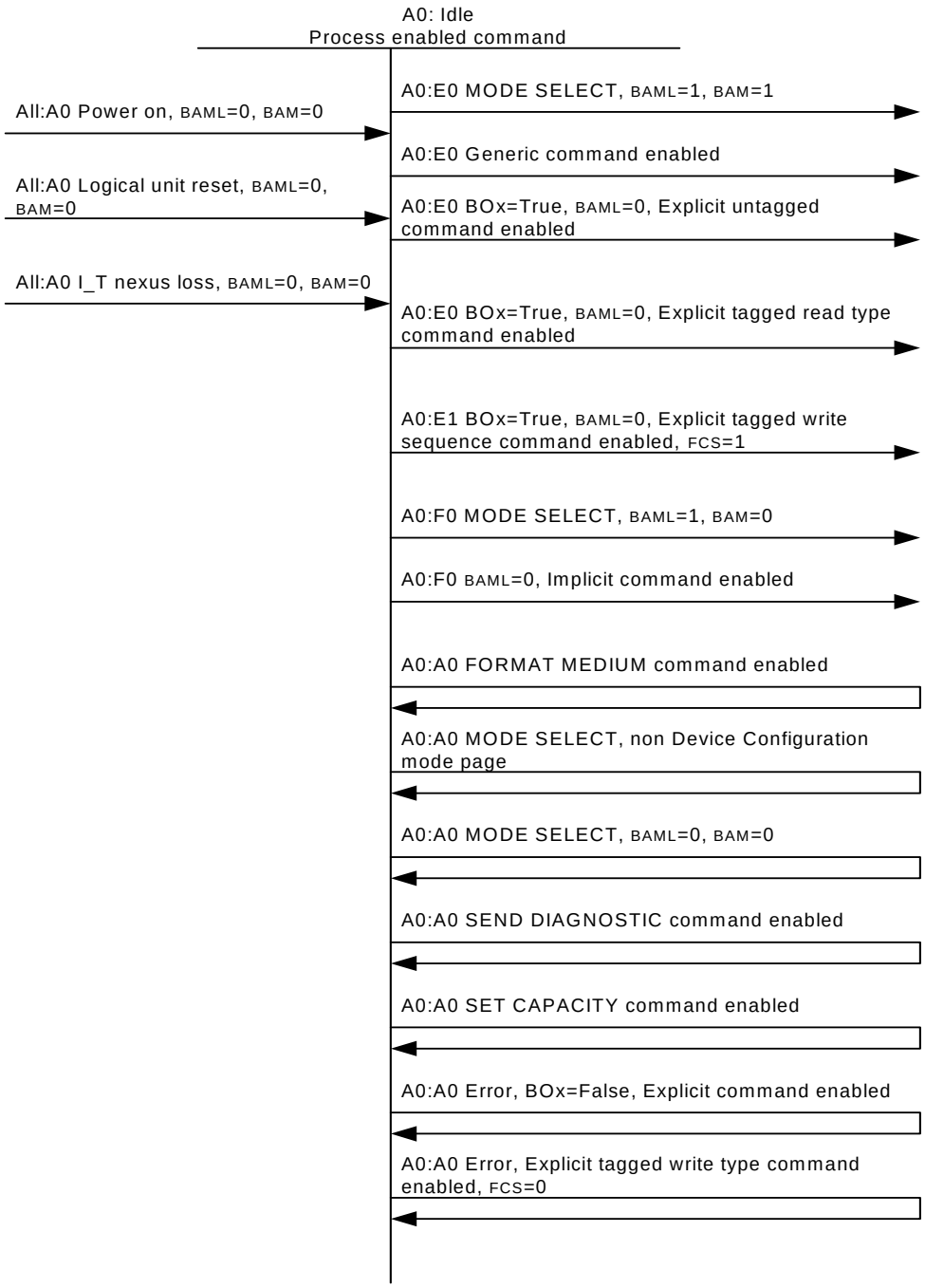

Figure 14 — Block address mode state diagram, Idle state

State A0:Idle: This is the idle state.

Transition All:A0: This transition shall occur when a power-on, logical unit reset, or I_T nexus loss event occurs and the BAML bit is set to zero and the BAM bit is set to zero.

Transition A0:E0: This transition shall occur when:

a) a MODE SELECT command specifying a Device Configuration mode page with the BAML bit set to one and the BAM bit set to one completes with GOOD status;
b) a generic command is enabled;
c) an explicit untagged command is enabled, the medium position is at BOx, and the BAML bit is set to zero; or
d) an explicit tagged read type command is enabled, the medium position is at BOx, and the BAML bit is set to zero.

Transition A0:E1: This transition shall occur when an explicit tagged write sequence command is enabled with the FCS bit set to one, the medium position is at BOx, and the BAML bit is set to zero.

Transition A0:F0: This transition shall occur when:

a) a MODE SELECT command specifying a Device Configuration mode page with the BAML bit set to one and the BAM bit set to zero completes with GOOD status; or
b) an implicit command is enabled, the medium position is at BOx, and the BAML bit is set to zero.

Transition A0:A0: This transition shall occur when:

a) a FORMAT MEDIUM command is enabled;
b) a MODE SELECT command specifying a mode page other than the Device Configuration mode page is enabled;
c) a MODE SELECT command specifying a Device Configuration mode page with the BAML bit set to zero and the BAM bit set to zero is enabled;
d) a SEND DIAGNOSTIC command is enabled;
e) a SET CAPACITY command is enabled;
f) an explicit command is enabled and the medium position is not at BOx. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to SEQUENTIAL POSITIONING ERROR; or
g) an explicit tagged write type command is enabled with the FCS bit set to zero. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB.

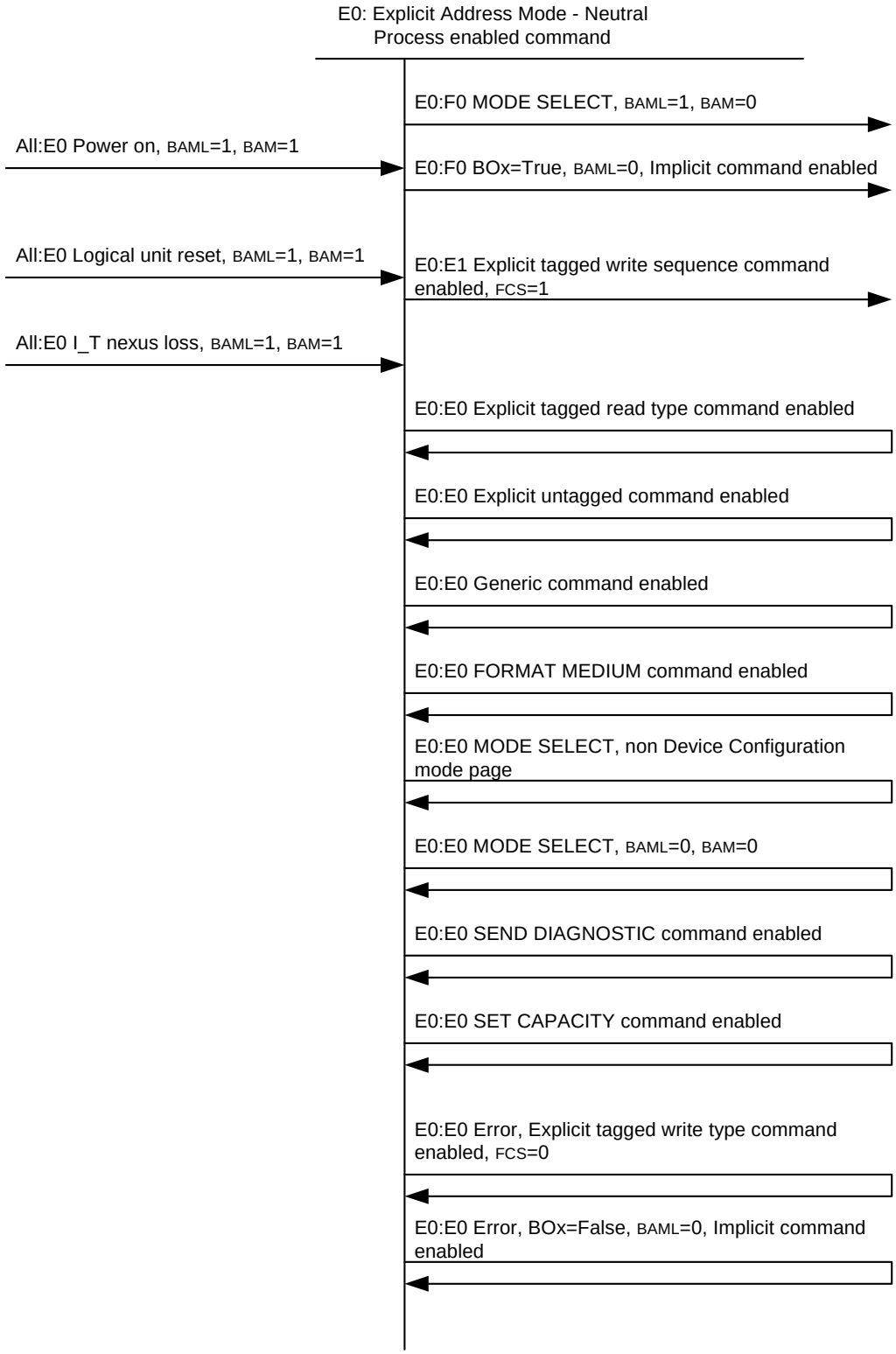

Figure 15 — Block address mode state diagram, Explicit Address Mode - Neutral

State E0:Explicit Address Mode - Neutral: This is the neutral state for explicit address mode.

Transition All:E0: This transition shall occur when a power-on, logical unit reset, or I_T nexus loss event occurs and the BAML bit is set to one and the BAM bit is set to one.

Transition E0:F0: This transition shall occur when:

a) a MODE SELECT command specifying a Device Configuration mode page with the BAML bit set to one and the BAM bit set to zero completes with GOOD status; or
b) an implicit command is enabled, the medium position is at BOx, and the BAML bit is set to zero.

Transition E0:E1: This transition shall occur when an explicit tagged write sequence command is enabled with the FCS bit set to one.

Transition E0:E0: This transition shall occur when:

a) an explicit tagged read type command is enabled;
b) an explicit untagged command is enabled;
c) a generic command is enabled;
d) a FORMAT MEDIUM command is enabled;
e) a MODE SELECT command specifying a mode page other than the Device Configuration mode page is enabled;
f) a MODE SELECT command specifying a Device Configuration mode page with the BAML bit set to zero and the BAM bit set to zero is enabled;
g) a SEND DIAGNOSTIC command is enabled;
h) a SET CAPACITY command is enabled;
i) an explicit tagged write type command is enabled with the FCS bit set to zero. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB; or
j) an implicit command is enabled, the medium position is not at BOx, and the BAML bit is set to zero. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to ILLEGAL COMMAND WHILE IN EXPLICIT ADDRESS MODE.

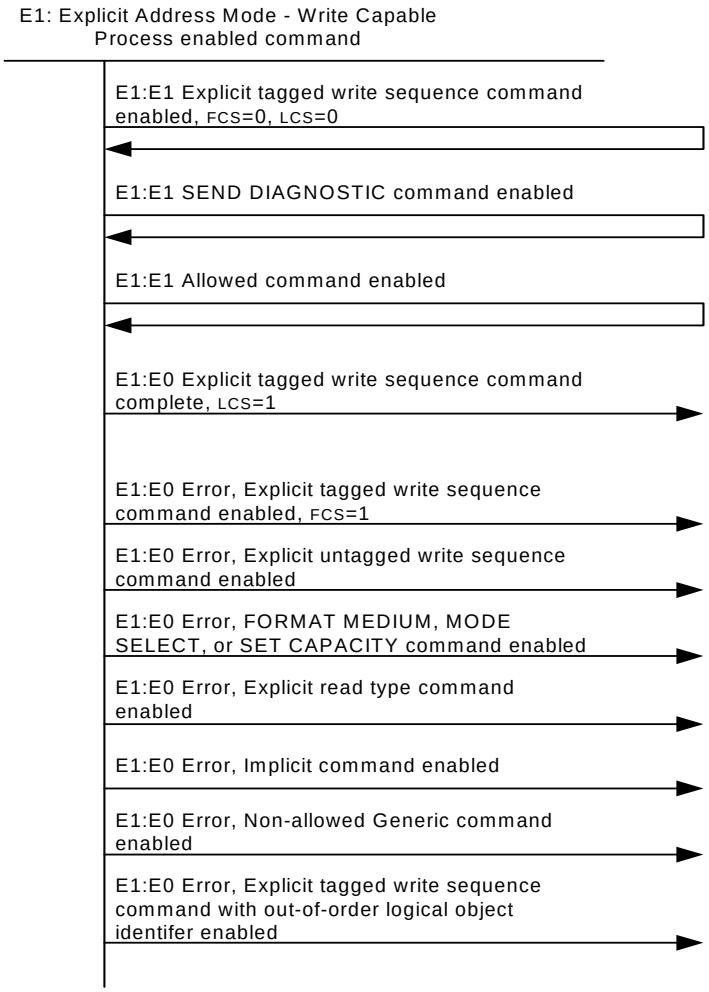

Figure 16 — Block address mode state diagram, Explicit Address Mode - Write Capable

State E1:Explicit Address Mode - Write Capable: This is the write capable state for explicit address mode.

Transition E1:E1: This transition shall occur when:

a) an explicit tagged write sequence command is enabled with the FCS bit set to zero and the LCS bit set to zero;
b) a SEND DIAGNOSTIC command is enabled; or
c) an allowed command (see table 21) is enabled.

Transition E1:E0: This transition shall occur when:

a) an explicit tagged write sequence command with the LCS bit set to one completed with GOOD status;
b) an explicit tagged write sequence command with the FCS bit set to one and the LCS bit set to one completed with GOOD status;
c) an explicit tagged write sequence command is enabled with the FCS bit set to one. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to INVALID FIELD IN CDB;

d) an explicit untagged write sequence command is enabled. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to ILLEGAL COMMAND WHILE IN WRITE CAPABLE STATE;
e) a FORMAT MEDIUM, MODE SELECT, or SET CAPACITY command is enabled. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to ILLEGAL COMMAND WHILE IN WRITE CAPABLE STATE;
f) an explicit read type command is enabled. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to ILLEGAL COMMAND WHILE IN WRITE CAPABLE STATE;
g) an implicit command is enabled. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to ILLEGAL COMMAND WHILE IN WRITE CAPABLE STATE;
h) a non-allowed generic command (see table 21) is enabled. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to ILLEGAL COMMAND WHILE IN WRITE CAPABLE STATE; or
i) an explicit tagged write sequence command with an out-of-order logical object identifier is enabled. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB.

Following an error condition that does not result in a transition out of write capable state, the following commands may be issued to transition from write capable state to neutral state:

a) a WRITE(16) command with the LCS bit set to one and the TRANSFER LENGTH field set to zero; or
b) a WRITE FILEMARKS(16) command with the LCS bit set to one, the IMMED bit set to zero, and the FILEMARK COUNT field set to zero.

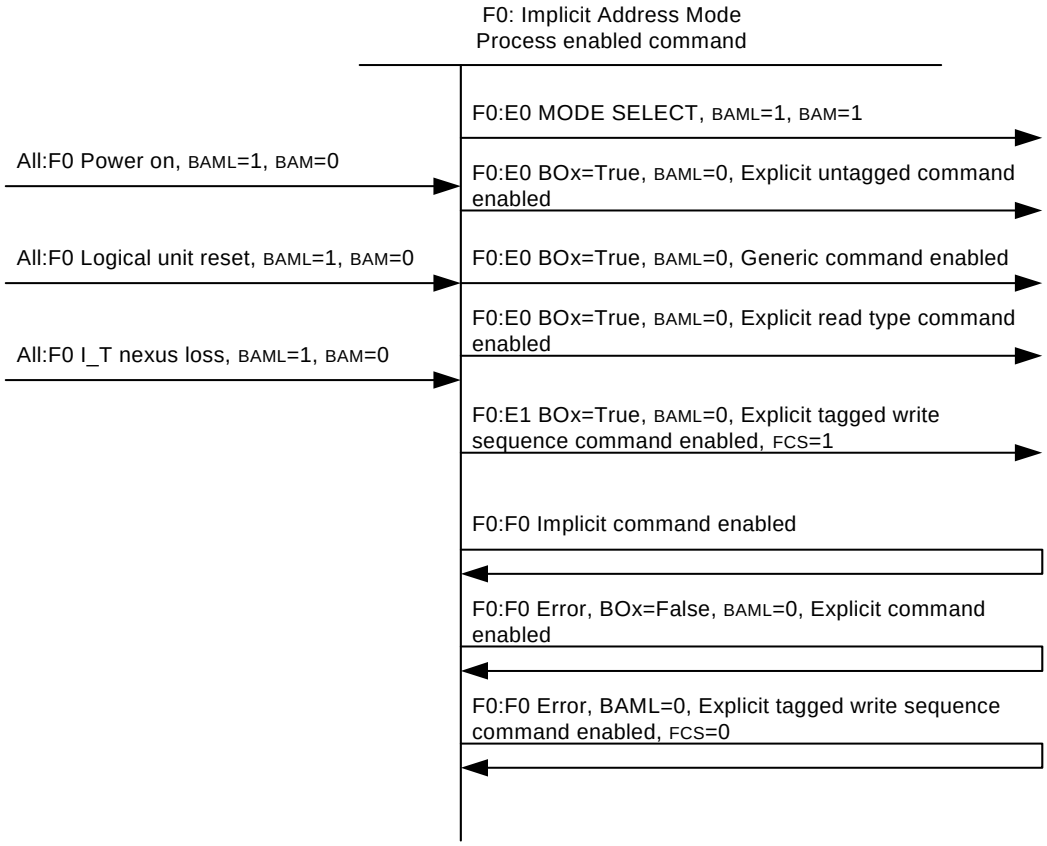

Figure 17 — Block address mode state diagram, Implicit Address Mode

State F0:Implicit Address Mode: This is the state for implicit address mode.

Transition All:F0: This transition shall occur when a power-on, logical unit reset, ot I_T nexus loss event occurs and the BAML bit is set to one and the BAM bit is set to zero.

Transition F0:E0: This transition shall occur when:

a) a MODE SELECT command specifying a Device Configuration mode page with the BAML bit set to one and the BAM bit set to one completes with GOOD status;
b) an explicit untagged command is enabled, the medium position is at BOx, and the BAML bit is set to zero;
c) a generic command is enabled, the medium position is at BOx, and the BAML bit is set to zero; or
d) an explicit read type command is enabled, the medium position is at BOx, and the BAML bit is set to zero.

Transition F0:E1: This transition shall occur when an explicit tagged write sequence command is enabled with the FCS bit set to one, the medium position is at BOx, and the BAML bit is set to zero.

Transition F0:F0: This transition shall occur when:

a) an implicit command is enabled;
b) an explicit command is enabled, the medium position is not at BOx, and the BAML bit is set to zero. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL

REQUEST, and the additional sense code shall be set to ILLEGAL COMMAND WHILE IN IMPLICIT ADDRESS MODE; or

c) an explicit tagged write sequence command is enabled with the FCS bit set to zero. In this case the device server shall return CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to ILLEGAL COMMAND WHILE IN IMPLICIT ADDRESS MODE.

4.2.17 TapeAlert application client interface

4.2.17.1 TapeAlert introduction

TapeAlert provides an application client with the capability to receive notification of various events and conditions arising in the SCSI target device. This standard defines 64 unique TapeAlert flags for a sequential-access device. A service information log parameter (see table 73) is also defined for each TapeAlert flag that provides information necessary for an application client to decide appropriate error recovery procedures.

TapeAlert flags fall into three categories of default severity (see table 9).

Table 9 — TapeAlert flags default severity

Value	Severity	Definition
01h	Informational	No guidance about continued operation without corrective action is given by this standard. The condition should be logged and/or the operator informed.
06h	Retryable	The event that generated this device information may be retried.
0Bh	Warning	The systme may not be operating optimally. Continued operation without corrective action may cause a failure or raise critical TapeAlert flags. The condition should be logged and/or the operator informed
10h	Critical	Either a failure has already occurred or a failure is imminent. Corrective action is required. The condition should be logged and/or an operator informed.
15h	Intervention required	If this condition is not corrected, a data loss failure may occur. The condition should be logged and/or an operator informed.
1Ah	Call service	Action by service personnel is required. The condition should be logged and service personnel informed.
All others	-	Reserved

Table 10 specifies the 64 TapeAlert flags for a sequential-access device. See Annex A for additional information about each TapeAlert flag.l

Table 10 — TapeAlert flags

Flag	Name	Type	Severity	Deactivation condition	TapeAlert Flag Specific Information available (see 8.2.3.x)
01h	Read warning	O	W	Start of next medium load	Y
02h	Write warning	O	W	Start of next medium load	Y
03h	Hard error	M	W	Start of next medium load[a]	N
04h	Media	M	C	Start of next medium load[a]	Y
05h	Read failure	M	C	Start of next medium load[a]	N
06h	Write failure	M	C	Start of next medium load[a]	N
07h	Media life	O	W	Start of next medium load	Y
08h	Not data grade	O	W	Start of next medium load	N
09h	Write protect	O	C	Start of next medium load or removal of write protect	N
0Ah	Media removal prevented	O	I	After medium removal allowed	N
0Bh	Cleaning media	O	I	Start of next medium load	N
0Ch	Unsupported format	O	I	Start of next medium load or format change	N
0Dh	Recoverable mechanical cartridge failure	O	C	Start of next medium load	N
0Eh	Unrecoverable mechanical cartridge failure	O	C	After service resolution	N
0Fh	Memory chip in cartridge failure	O	W	Start of next medium load	N
10h	Forced eject	O	C	Start of next medium load	N
11h	Read only format	O	W	Start of next medium load or format change	N
12h	Tape directory corrupted on load	O	W	Start of next medium load	N
13h	Nearing media life	O	I	Start of next medium load	Y
14h	Cleaning required	O	C	After successful cleaning or cause resolved	Y
15h	Cleaning requested	O	W	After successful cleaning	Y
Type Key: M=Mandatory O=Optional					
a) Devices compliant with previous versions of this standard may deactivate this TapeAlert flag when demounting the current medium.					

Table 10 — TapeAlert flags (Continued)

Flag	Name	Type	Severity	Deactivation condition	TapeAlert Flag Specific Information available (see 8.2.3.x)
16h	Expired cleaning media	O	C	Start of next medium load	Y
17h	Invalid cleaning tape	O	C	Start of next medium load	
18h	Retension requested	O	W	After successful retention	N
19h	Multi-port interface error on a primary port	O	W	After interface returns to operation	N
1Ah	Cooling fan failure	O	W	After service resolution	N
1Bh	Power supply failure	O	W	After service resolution	N
1Ch	Power consumption	O	W	After power consumption returns to within specification	Y
1Dh	Drive preventive maintenance required	O	W	After service resolution	N
1Eh	Hardware A	O	C	After service resolution	N
1Fh	Hardware B	M	C	At power on event	N
20h	Primary interface	O	W	After interface returns to operation	N
21h	Eject media	O	C	Start of next medium load	N
22h	Microcode update fail	O	W	Start of next microcode update	N
23h	Drive humidity	O	W	After humidity returns to within specification	Y
24h	Drive temperature	O	W	After temperature returns to within specification	Y
25h	Drive voltage	O	W	After voltage returns to within specification	Y
26h	Predictive failure	O	C	After service resolution	Y
27h	Diagnostics required	O	W	After service resolution	N
28h - 2Eh	Obsolete				N
2Fh - 31h	Reserved				N
32h	Lost statistics	O	W	Start of next medium load	N
Type Key: M=Mandatory O=Optional					
a) Devices compliant with previous versions of this standard may deactivate this TapeAlert flag when demounting the current medium.					

Table 10 — TapeAlert flags (Continued)

Flag	Name	Type	Severity	Deactivation condition	TapeAlert Flag Specific Information available (see 8.2.3.x)
33h	Tape directory invalid at unload	O	W	Start of next medium load	N
34h	Tape system area write failure	O	C	Start of next medium load	N
35h	Tape system area read failure	O	C	Start of next medium load	N
36h	No start of data	O	C	Start of next medium load	N
37h	Loading or threading failure	O	C	Start of next medium load	N
38h	Unrecoverable unload failure	O	C	After service resolution	N
39h	Automation interface failure	O	C	After service resolution	N
3Ah	Microcode failure	O	W	After service resolution	N
3Bh	WORM Medium - Integrity Check Failed	O	W	Start of next medium load	N
3Ch	WORM Medium - Overwrite Attempted	O	W	Start of next medium load	N
3Dh - 40h	Reserved				
Type Key: M=Mandatory O=Optional					
a) Devices compliant with previous versions of this standard may deactivate this TapeAlert flag when demounting the current medium.					

4.2.17.2 TapeAlert usage model

4.2.17.2.1 TapeAlert usage model introduction

This standard specifies three methods for an application client to monitor activation of TapeAlert flags:

a) polling either the TapeAlert log page or the TapeAlert Response log page;
b) configuring the device server to establish an Informational exception condition upon activation of one or more TapeAlert flags; and
c) establishing a threshold for one or more of the parameters in the TapeAlert log page.

An application client may use any of these methods or a mixture of them.

Prior to using the TapeAlert Response log page with method (a), an application client should determine whether the device server supports the TapeAlert Response log page. An application client may determine if a device server supports a log page by issuing a LOG SENSE command with the PAGE CODE field set to 00h and examining the data returned.

4.2.17.2.2 TapeAlert polling usage model

The application client configures the device server for the TapeAlert polling usage model by:

a) setting the TASER bit in the Device Configuration Extension mode page to one (see 8.3.8); and
b) setting the ETC bit of every parameter in the TapeAlert log page to zero (see 8.2.3).

NOTE 4 Devices that comply with earlier versions of this standard set the ETC bit in each TapeAlert log parameter to zero and do not allow the application client to change this value.

If using the TapeAlert polling usage model, the application client reads the TapeAlert log page or the TapeAlert Response log page without receiving notification from the device server that a TapeAlert flag has changed state. The application client may read the TapeAlert log page or the TapeAlert Response log page at any time (e.g. polled at a regular interval of 60 seconds). The application client should read either the TapeAlert log page or the TapeAlert Response log page:

a) priot to mounting a volume and at the beginning of a data transfer sequence;
b) immediately after detecting an unrecoverable error during the data transfer sequence;
c) before demounting each volume; and
d) at the end of a data transfer sequence.

4.2.17.2.3 TapeAlert informational exception usage model

The application client configures the device server for the TapeAlert informational exception usage model by:

a) setting the TASER bit in the Device Configuration Extension mode page to zero (see 8.3.8);
b) setting the DEXCPT bit in the Informational Exceptions Control mode page to zero and the test bit in the Informational Exceptions Control mode page to zero (see 8.3.6);
c) setting the MRIE field in the Informational Exceptions Control mode page to a supported value greater than zero (see 8.3.6); and
d) setting the ETC bit of every parameter in the TapeAlert log page to zero (see 8.2.3).

NOTE 5 Devices that comply with earlier versions of this standard set the ETC bit in each TapeAlert log parameter to zero and do not allow the application client to change this value.

If using the TapeAlert informational exception usage model, the application client receives TapeAlert flag information upon or after receiving notification from the device server that an informational exception has occurred. The device server generates an informational exception condition due to an activated TapeAlert flag. The device server does not generate an informational exception condition due to a de-activated TapeAlert flag. The method used by the device server to report the informational exception condition depends on the value of the MRIE field (see SPC-4). If the device server returns descriptor format sense data (see SPC-4), the current state of all TapeAlert flags appears in the Information sense data descriptor (see 4.2.12.3). If the device server returns fixed format sense data (see SPC-4), the application client should read the TapeAlert log page to retrieve the state of the TapeAlert flags.

If the TEST bit is set to zero, a device server reporting an informational exception condition for a TapeAlert flag sets the additional sense code to FAILURE PREDICTION THRESHOLD EXCEEDED.

4.2.17.2.4 TapeAlert threshold usage model

The application client configures the device server for the TapeAlert threshold usage model by:

a) setting the TASER bit in the Device Configuration Extension mode page to one (see 8.3.8);

b) setting to one the ETC bit of each parameter in the TapeAlert log page for which the application client wishes to receive a unit attention condition (see 8.2.3);
c) setting to zero the ETC bit of each parameter in the TapeAlert log page for which the application client does not wish to receive a unit attention condition (see 8.2.3)
d) establishing a threshold value and a threshold met criteria (tmc) value for each TapeAlert log page parameter with the etc bit set to one (see SPC-4).

NOTE 6 Devices that comply with earlier versions of this standard set the ETC bit in each TapeAlert log parameter to zero and do not allow the application client to change this value. These devices do not support the TapeAlert threshold usage model.

If using the TapeAlert threshold usage model, the application client receives a unit attention when a TapeAlert log page parameter meets its threshold criteria. If the device server returns descriptor format sense data (see SPC-4), the current state of all TapeAlert flags appears in the Information sense data descriptor (see 4.2.12.3). If the device server returns fixed format sense data (see SPC-4), the application client should read the TapeAlert log page to retrieve the state of the TapeAlert flags.

The threshold and TMC values determine whether the device server generates a unit attention condition on TapeAlert flag activation or de-activation.

4.2.17.3 TapeAlert flag activation and deactivation

The device server shall activate a TapeAlert flag upon detecting the condition or event specified in table 11.

Table 11 — TapeAlert flag activation conditions

Flag	Name	Activation condition
03h	Hard error	An unrecoverable read/write/positioning error.
04h	Media	An unrecoverable read/write/positioning error due to a faulty medium.
05h	Read failure	An unrecoverable read error due to either a faulty medium or faulty device hardware.
06h	Write failure	An unrecoverable write/positioning error due to either a faulty medium or faulty device hardware.
1Fh	Hardware B	An error during power on self test.

The device server may activate a TapeAlert flag not listed in table 11 upon detection of a vendor-specific condition.

Initialization processing due to a power-on condition may activate some TapeAlert flags.

The device server shall deactivate a TapeAlert flag upon detecting the condition or event specified for that flag in table 10. The device server shall not deactivate any TapeAlert flag due to a vendor-specific condition/event. The device server shall deactivate all TapeAlert flags:

a) upon processing a LOG SENSE command with the PAGE CODE field set to 2Eh if the TAPLSD bit is set to zero in the Device Configuration Extension mode page (see 8.3.8); or
b) upon detecting a logical unit reset condition (see SAM-4).

The device server may deactivate any TapeAlert flag on a vendor-specific basis due to:

a) processing a LOG SELECT command with the PCR field set to one (see SPC-4); or
b) processing a LOG SELECT command with the PARAMETER LIST LENGTH field set to zero and the PC field set to 11b.

If the device server deactivates a TapeAlert flag by processing a LOG SENSE command with the PAGE CODE field set to 2Eh, the device server shall not activate the flag again until the device server:

a) detects the deactivation condition specified in table 10;
b) detects a logical unit reset condition; or
c) processes a LOG SELECT command with the PCR field set to one.

If the device server deactivates a TapeAlert flag through some other mechanism, the device server may activate the flag before

a) detecting the deactivation condition given in table 10;
b) detecting a logical unit reset condition; or
c) processing a LOG SELECT command with the PCR field set to one.

If the TAPLSD bit in the Device Configuration Extension mode page (see 8.3.8) is set to zero, the device server should deactivate flags on a per I_T nexus basis such that active flags are available for reading by other I_T nexuses. If the TAPLSD bit in the Device Configuration Extension mode page (see 8.3.8) is set to one, the device server may deactivate flags on a per I_T nexus basis.

NOTE 7 The device server deactivating TapeAlert flags on any basis other than per I_T nexus, if the TAPLSD bit is set to zero, violates backwards compatibility with previous versions of this standard.

4.2.17.4 WORM TapeAlert flags

Two TapeAlert flags exist to support Write Once Read Many (WORM) media:

a) 3Bh, WORM Medium, Integrity Check Failed; and
b) 3Ch, WORM Medium, Overwrite Attempted.

If the device server supports TapeAlert flag 3Bh, it shall activate that flag upon detecting that the integrity of the medium may be compromised. If the device server supports TapeAlert flag 3Ch, it shall activate that flag if an application client attempts to overwrite or erase user data on an archive tape (see 4.2.20).

In addition to the deactivation conditions for all TapeAlert flags (see 4.2.17.3), the device server shall activate TapeAlert flags 3Bh and 3Ch upon:

a) execution of a LOAD UNLOAD command with a load bit set to one (see 7.2) that results in a not ready to ready transition, or when both the medium and device server support MAM, that results in access to medium auxiliary memory only; or

execution of an autoload operation (see SPC-4) that results in a not ready to ready transition, or when both the medium and device server support MAM, that results in access to medium auxiliary memory only.

4.2.17.5 TapeAlert Response log page

The TapeAlert flags reported through the TapeAlert Response log page (see 8.2.1) represent states. This approach facilitates accurate reporting of the conditions encountered by the device and allows the application client to manage the information directly. The device server does not maintain unique TapeAlert information for each I_T nexus, and the state flags are not affected by port events (e.g., I_T nexus loss).

The application client is responsible for determining which flags have changed state upon subsequent retrieval of the TapeAlert Response log page, requiring the application client to maintain at least one previously retrieved TapeAlert Response log page in order to detect differences. The application client may maintain a state change history.

If the TAPLSD bit in the Device Configuration Extension mode page (see 8.3.8) is set to one, the device server shall maintain the value of the flags in the TapeAlert Response log page independently of the TapeAlert flags reported through the TapeAlert log page (see 8.2.3). A LOG SENSE command that retrieves the TapeAlert Response log page shall not set the flags in that page to zero and shall not set the flags in the TapeAert log page to zero. A LOG SENSE command that retrieves the TapeAlert log page shall not set the flags in the TapeAlert Response log page to zero.

4.2.18 READ ATTRIBUTE and WRITE ATTRIBUTE command support

Support for the READ ATTRIBUTE and WRITE ATTRIBUTE commands (see SPC-4) is described in table 12 and table 13.

Table 12 — Device common attributes

ID	Attribute Name	Number of Bytes	Format
0002h	TAPEALERT FLAGS	8	Binary
0005h	ASSIGNING ORGANIZATION	8	ASCII
0006h	FORMATTED DENSITY CODE	1	Binary

The TAPEALERT FLAGS attribute provides a means of reporting the state of the TapeAlert flags for the previous load of the medium. Each TapeAlert flag occupies one bit (Flag 1 = MSB, Byte 1; Flag 64 = LSB, Byte 8). The bits specify all the TapeAlert flags that were set during the previous load, (i.e., the bits are “sticky” for the load).

The ASSIGNING ORGANIZATION attribute identifies the organization responsible for the specifications defining the values in the FORMATTED DENSITY CODE attribute. The ASSIGNING ORGANIZATION attribute should contain a vendor identification. The use of specific vendor identification other than the one associated with the device is allowed.

NOTE 8 It is intended that the ASSIGNING ORGANIZATION attribute provide a unique vendor identification of the FORMATTED DENSITY CODE attribute. In the absence of a formal registration procedure, T10 maintains a list of known vendor identification codes for use in the Standard INQUIRY data (see SPC-4). Vendors are requested to voluntarily submit their identification codes to T10 to prevent duplication of codes (see SPC-4).

If the device server formats the medium using a format other than the one specified in the MEDIUM DENSITY CODE attribute (e.g., for compatibility with a previous generation format), then the FORMATTED DENSITY CODE attribute specifies the DENSITY CODE of the format chosen (see SPC-4). Otherwise this attribute shall be the same as the MEDIUM DENSITY CODE attribute.

Table 13 — Medium common attributes

ID	Attribute Name	Number of Bytes	Format
0402h	MEDIUM LENGTH	4	Binary
0403h	MEDIUM WIDTH	4	Binary
0404h	ASSIGNING ORGANIZATION	8	ASCII
0405h	MEDIUM DENSITY CODE	1	Binary

The MEDIUM LENGTH attribute specifies the length of the medium in meters. A value of 0h specifies that the length of the medium is undefined.

The MEDIUM WIDTH attribute specifies the width of the medium supported by this density. This attribute has units of tenths of millimeters. The value in this attribute shall be rounded up if the fractional value of the actual value is

greater than or equal to 0,5. The MEDIUM WIDTH attribute may vary for a given density depending on the mounted medium. A value of 0h specifies the width of the medium is undefined.

The ASSIGNING ORGANIZATION attribute identifies the organization responsible for the specifications defining the values in the MEDIUM DENSITY CODE attribute. The ASSIGNING ORGANIZATION attribute should contain a vendor identification.

NOTE 9 It is intended that the ASSIGNING ORGANIZATION attribute provide a unique vendor identification of the MEDIUM DENSITY CODE attribute. In the absence of a formal registration procedure, T10 maintains a list of known vendor identification codes for use in the Standard INQUIRY data (see SPC-4). Vendors are requested to voluntarily submit their identification codes to T10 to prevent duplication of codes (see SPC-4).

4.2.19 Reservations

Reservation restrictions are placed on commands as a result of access qualifiers associated with the type of reservation. See SPC-4 for a description of reservations. The details of which commands are allowed under what types of reservations are described in table 14.

Commands from I_T_nexuses holding a reservation should complete normally. Table 14 specifies the behavior of commands from registered I_T_nexuses when a registrants only or all registrants persistent reservation is present.

NOTE 10 Due to the nature of streaming device types, Write Exclusive and Write Exclusive, Registrants Only modes of reservation do not protect an application client's continuity of operations when using the implicit address command set. While these modes do protect unauthorized modification of data, they do not protect from medium position changes that may result in errors due to incorrect position. It is the responsibility of the application client to manage this using means beyond the scope of this specification. Application clients should use exclusive modes of reservation while accessing the medium to prevent interference from other applications.

For each command, this standard and the SPC-4 standard define the conditions that result in RESERVATION CONFLICT.

Table 14 — SSC-3 commands that are allowed in the presence of various reservations

Command	Addressed LU has this type of persistent reservation held by another I_T_nexus				
	From any I_T_nexus		From registered I_T_nexus (RR all types)	From I_T_nexus not registered	
	Write Exclusive	Exclusive Access		Write Exclusive - RR	Exclusive Access - RR
ERASE(6)	Conflict	Conflict	Allowed	Conflict	Conflict
ERASE(16)	Conflict	Conflict	Allowed	Conflict	Conflict
FORMAT MEDIUM	Conflict	Conflict	Allowed	Conflict	Conflict
LOAD UNLOAD	Conflict	Conflict	Allowed	Conflict	Conflict
LOCATE(10)	Allowed	Conflict	Allowed	Allowed	Conflict
LOCATE(16)	Allowed	Conflict	Allowed	Allowed	Conflict
READ(6)	Allowed	Conflict	Allowed	Allowed	Conflict
READ(16)	Allowed	Conflict	Allowed	Allowed	Conflict
READ BLOCK LIMITS	Allowed	Allowed	Allowed	Allowed	Allowed

Table 14 — SSC-3 commands that are allowed in the presence of various reservations (Continued)

Command	Addressed LU has this type of persistent reservation held by another I_T_nexus				
	From any I_T_nexus		From registered I_T_nexus (RR all types)	From I_T_nexus not registered	
	Write Exclusive	Exclusive Access		Write Exclusive - RR	Exclusive Access - RR
READ POSITION	Allowed	Conflict	Allowed	Allowed	Conflict
READ REVERSE(6)	Allowed	Conflict	Allowed	Allowed	Conflict
READ REVERSE(16)	Allowed	Conflict	Allowed	Allowed	Conflict
RECOVER BUFFERED DATA	Allowed	Conflict	Allowed	Allowed	Conflict
REPORT DENSITY SUPPORT	Allowed	Allowed	Allowed	Allowed	Allowed
REWIND	Allowed	Conflict	Allowed	Allowed	Conflict
SET CAPACITY	Conflict	Conflict	Allowed	Conflict	Conflict
SPACE(6)	Allowed	Conflict	Allowed	Allowed	Conflict
SPACE(16)	Allowed	Conflict	Allowed	Allowed	Conflict
VERIFY(6)	Allowed	Conflict	Allowed	Allowed	Conflict
VERIFY(16)	Allowed	Conflict	Allowed	Allowed	Conflict
WRITE(6)	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE(16)	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE FILEMARKS(6)	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE FILEMARKS(16)	Conflict	Conflict	Allowed	Conflict	Conflict

Key: **LU**=Logical Unit, **RR**=Registrants Only or All Registrants

Allowed: Commands received from I_T nexuses not holding the reservation or from I_T nexuses not registered when a registrants only or all registrants type persistent reservation is present should complete normally.

Conflict: Commands received from I_T nexuses not holding the reservation or from I_T nexuses not registered when a registrants only or all registrants type persistent reservation is present shall not be performed and the device server shall terminate the command with RESERVATION CONFLICT status.

4.2.20 Archive tape and WORM mode

4.2.20.1 WORM overview

The device server may support a Write Once, Read Many (WORM) mode of operation. This mode of operation places additional restrictions on the processing of commands by the devices server.

4.2.20.2 Archive tape

An archive tape is identified by a format-defined method (e.g., the Medium Type attribute in the MAM data returning a Write Once Medium type value).

4.2.20.3 WORM mode

A device server that supports WORM mode and detects an archive tape is mounted shall enter WORM mode. While in WORM mode, WRITE, WRITE FILEMARKS, ERASE, FORMAT MEDIUM, SET CAPACITY, and MODE SELECT commands that alter the partitioning or format are subject to the restrictions applied to archive tape by the device server (see 8.3.7).

If a device server operating in WORM mode detects that one of these additional restrictions would be violated by a command being processed, the device server shall terminate the command with CHECK CONDITION status. The sense key shall be set to DATA PROTECT and the additional sense code shall be set to WORM MEDIUM - OVERWRITE ATTEMPTED. If a write protection is in effect for the device server or medium (see 4.2.13), it shall take precedence over the WORM mode violation.

If a device server operating in WORM mode is unable to determine if medium is such that one of these additional restrictions would be violated by a command being processed, the device server shall terminate the command with CHECK CONDITION status. The sense key shall be set to DATA PROTECT and the additional sense code shall be set to WORM MEDIUM - OVERWRITE ATTEMPTED. If a write protection is in effect for the device server or medium (see 4.2.13), it shall take precedence over the WORM mode violation.

If a device server that does not support WORM mode detects that an archive tape is mounted, it shall treat the medium as write protected. Any command that attempts to alter the medium shall be terminated with CHECK CONDITION status. The sense key shall be set to DATA PROTECT and the additional sense code shall be set to CANNOT WRITE MEDIUM - INCOMPATIBLE FORMAT.

4.2.21 Data encryption

4.2.21.1 Data encryption overview

A device compliant with this standard may contain hardware or software that is capable of encrypting and decrypting the data within logical blocks to provide security against unauthorized access to that data. The SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT commands specifying the Tape Data Encryption security protocol (see 8.5.2 and 8.5.3) provide a means for the application client to monitor and control the encryption and decryption processes within the device server. A device server that supports the SECURITY PROTOCOL OUT command shall also support the SECURITY PROTOCOL IN command.

The SECURITY PROTOCOL OUT command specifying the Tape Data Encryption security protocol is used to set data encryption parameters. The SECURITY PROTOCOL IN command specifying the Tape Data Encryption security protocol is used to discover the type of data security features supported by the device server, the current configuration of data security features, and status of the encryption and decryption processes.

4.2.21.2 Encrypting data on the medium

The application client controls the data encryption process by use of the SECURITY PROTOCOL OUT command specifying the Tape Data Encryption security protocol. Data encryption shall be managed within the device server on a per I_T_L nexus basis. The data encryption process is enabled for an I_T_L nexus upon successful completion of a SECURITY PROTOCOL OUT command that sends a Set Data Encryption page (see 8.5.3.2) with the ENCRYPTION MODE field set to ENCRYPT and with a valid key. If the data encryption scope parameter for an I_T_L nexus is set to PUBLIC (see 4.2.21.7), the data encryption process may be enabled by another I_T_L nexus that establishes a set of data encryption parameters with a key scope of ALL I_T NEXUS (see 4.2.21.8).

If data encryption is enabled for an I_T_L nexus and the mounted volume supports the selected encryption algorithm at the current logical position, all logical blocks received by the device server from that I_T_L nexus as part of a WRITE(6) or WRITE(16) command shall be encrypted before being recorded on the medium. Filemarks are logical objects that shall not be encrypted.

If data encryption is enabled for an I_T_L nexus and the mounted volume does not support the selected encryption algorithm at the current logical position, then the device server shall terminate a WRITE(6) or WRITE(16) command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to ENCRYPTION PARAMETERS NOT USEABLE.

If data encryption is enabled for an I_T_L nexus and the mounted volume does not support the selected encryption algorithm at the current logical position, then the device server may terminate a WRITE FILEMARKS(6) or WRITE FILEMARKS(16) command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to ENCRYPTION PARAMETERS NOT USEABLE.

4.2.21.3 Reading encrypted blocks on the medium

A volume may contain no encrypted blocks, all encrypted blocks, or a mixture of encrypted blocks and unencrypted blocks. The fact that logical blocks are encrypted shall not alter space or locate operations. The decryption mode shall be ignored when processing a filemark during a read or verify operation.

A device server that supports encryption should be capable of distinguishing encrypted blocks from unencrypted blocks. The device server reports its capability of distinguishing encrypted blocks from unencrypted blocks through the DED_C bit in the Data Encryption Algorithm descriptor (see 8.5.2.4). If the device server is capable of distinguishing encrypted blocks from unencrypted blocks, an attempt to read or verify an encrypted block when the decryption mode is set to DISABLED shall cause the device server to terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to UNABLE TO DECRYPT DATA. The device server shall establish the logical position at the BOP side of the encrypted block.

If the device server is capable of distinguishing encrypted blocks from unencrypted blocks and the decryption mode is set to DECRYPT or RAW, an attempt to read or verify an unencrypted block shall cause the device server to terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to UNENCRYPTED DATA ENCOUNTERED WHILE DECRYPTING. The device server shall establish the logical position at the BOP side of the unencrypted block.

> NOTE 11 It is possible for a device server that is not capable of distinguishing encrypted blocks from unencrypted blocks to decrypt data that was not encrypted. Application clients are responsible for checking the integrity of the data in this environment.

A device server that supports encryption and has been configured to decrypt the data may be capable of determining that the encryption key is correct for an encrypted block. The correct key to use for this encrypted block may be either the encryption key or one of the supplemental decryption keys (SDK). The method for determining which key to use for decryption shall be vendor specific. If the device server is capable of determining that the encryption key is correct, an attempt to read or verify an encrypted block when the decryption mode is set to either DECRYPT or MIXED, but all of the keys provided are incorrect for the encrypted block, shall cause the device server to terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to INCORRECT DATA ENCRYPTION KEY. The device server shall establish the logical position at the BOP side of the encrypted block.

A device server that supports encryption and has been configured to decrypt the data may be capable of validating the integrity of the data after decrypting it (i.e., that the decrypted data matches the data that was encrypted). If the device server is capable of validating the integrity of the data after decrypting it, an attempt to read or verify an encrypted block when the decryption mode is set to either DECRYPT or MIXED but the data fails the integrity validation process shall cause the device server to terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to CRYPTOGRAPHIC INTEGRITY VALIDATION FAILED. The device server shall establish the logical position at the BOP side the encrypted block.

A device server that is capable of distinguishing encrypted blocks from unencrypted blocks and has been configured to decrypt the data should perform at least one of the following for each encrypted block that is decrypted:

a) determine if the encryption key is correct for the encrypted block; or
b) validate the integrity of the data after decrypting it.

A device server that is capable of both determining if the encryption key or one of the supplemental decryption keys is correct for the encrypted block and validating the integrity of the data after decrypting it shall:

1) determine if the encryption key is correct for the encrypted block; and
2) validate the integrity of the data.

4.2.21.4 Exhaustive-search attack prevention

To prevent an exhaustive-search attack from discovering the encryption key or one of the supplemental decryption keys, the device server should provide a mechanism to prevent unlimited attempts at setting a data encryption key or supplemental decryption keys and then attempting to read the data. The use of such a mechanism may protect against an encryption algorithm being broken due to undiscovered mathematical weaknesses in the encryption algorithm.

If the device server has reached its limit on failed attempts to set the data encryption key or supplemental decryption keys and decrypt data, it shall disable decryption for all I_T nexuses. All subsequent SECURITY PROTOCOL OUT commands specifying the Tape Data Encryption security protocol and with the SECURITY PROTOCOL SPECIFC field set to Set Data Encryption page with the DECRYPT field or ENCRYPT field set to any value other than DISABLE shall be terminated with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to DATA DECRYPTION KEY FAIL LIMIT REACHED. This condition shall persist until the volume is demounted or a hard reset condition occurs.

4.2.21.5 Keyless copy of encrypted data

In some scenarios it is desirable to copy data from one volume to another without needing knowledge of the encryption parameters used to encrypt the data on the volume.

A keyless copy logical unit (KCLU) controls configuration and data flows related to a volume that is either a source or destination for encrypted data being transferred without requiring application client knowledge of an encryption key.

A keyless copy source logical unit (KCSLU) controls configuration and data flows related to the volume from which the encrypted data is copied without requiring device server knowledge of an encryption key when the decryption mode is set to RAW.

A keyless copy destination logical unit (KCDLU) controls configuration and data flows related to the volume to which the encrypted data is being copied without requiring device server knowledge of an encryption key when the encryption mode is set to EXTERNAL.

To accomplish a keyless copy operation an application client sets the KCSLU decryption mode to RAW and the KCDLU encryption mode to EXTERNAL. The application client then reads one or more logical objects from the KCSLU and writes those logical objects to the KCDLU. During this process, if the KCSLU detects a mismatch between the key-associated data in the data encryption parameters and the key-associated data on the medium during a read operation, then the KCSLU returns a CHECK CONDITION status to the application client to notify it that some action is required. An example of this is shown in the informative flowchart in Annex C.

It shall not be considered an error if a Set Data Encryption page has the DECRYPTION MODE field is set to RAW and any of the key-associated data descriptors required by the algorithm specified by the value in the ALGORITHM INDEX field are not present.

If the encryption algorithm in use by the KCSLU requires key-associated data to be included in the Set Data Encryption page when the ENCRYPTION MODE field is set to EXTERNAL, then an attempt to read or verify an encrypted block while the decryption mode is set to RAW shall cause the KCSLU to compare all key-associated data associated with each encrypted block that is read or verified to the corresponding key-associated data that are part of the current encryption parameters. Key-associated data required to be compared by the decryption algorithm that do not match or are not present shall cause the KCSLU to terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to INCORRECT ENCRYPTION PARAMETERS. The KCSLU shall establish the logical position at the BOP side of the block.

If a KCDLU receives a SECURITY PROTOCOL OUT command with a Set Data Encryption page with the ENCRYPTION MODE field set to EXTERNAL, and any of the key associated data descriptors required by the data encryption algorithm specified by the ALGORITHM INDEX field are not present or are not supported, then the KCDLU shall terminate the command with CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

Some encryption algorithms provide a mechanism to record with encrypted blocks the encryption mode setting when the encrypted block was written. The device server reports if an encryption algorithm supports this mechanism by way of the EAREM bit in the algorithm descriptor (see 8.5.2.4). If the encryption algorithm provides this capability, the device server may support a feature to check during read and verify operations if the block was written with the encryption mode set to EXTERNAL. The CEEM field in the Set Data Encryption page (see 8.5.3.2) provides the means to control the process of checking the encryption mode used when an encrypted block was written to tape. If the decryption mode is set to DECRYPT or MIXED and the check external encryption mode data encryption parameter (see 8.5.3.2) is set to 10b:

1) the device server shall verify that each encrypted block that is processed for read and verify commands was written with the encryption mode set to ENCRYPT; and
2) if an attempt is made to read or verify an encrypted block that was written with the encryption mode set to EXTERNAL, the device server shall terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT and the additional sense key set to ENCRYPTION MODE MISMATCH ON READ.

If the decryption mode is set to DECRYPT or MIXED and the check external encryption mode data encryption parameter is set to 11b:

1) the device server shall verify that each encrypted block that is processed for read and verify commands was written with the encryption mode set to EXTERNAL; and
2) if an attempt is made to read or verify an encrypted block that was written with the encryption mode set to ENCRYPT, then the device server shall terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT and the additional sense key set to ENCRYPTION MODE MISMATCH ON READ.

The check external encryption mode data encryption parameter shall not affect space or locate operations. The check external encryption mode data encryption parameter shall not affect read or verify operations on filemarks and unencrypted blocks.

Some encryption algorithms provide a mechanism to record with encrypted blocks an indication that they are disabled for raw decryption mode operations. The device server reports if an encryption algorithm supports this

mechanism by way of the RDMC_C field in the algorithm descriptor (see 8.5.2.4). If the decryption mode is set to RAW and the encryption algorithm supports this feature, then:

1) the device server shall check the format specific indication that disables raw decryption mode operations for each encrypted block that is processed for read and verify commands; and
2) if an attempt is made to read or verify an encrypted block that was disabled for raw decryption mode operations, then the device server shall terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT and the additional sense key set to ENCRYPTED BLOCK NOT RAW READ ENABLED.

4.2.21.6 Managing keys within the physical device

To increase the security of keys, the data encryption parameters are volatile in the physical device and the data encryption keys are never reported to an application client. The physical device also may have limited resources for storage of keys.

A device server that supports encryption shall support at least one of the key formats that are defined in this standard (see table 147).

A vendor-specific key reference is an identifier that is associated with a specific key. The method by which keys and their associated vendor-specific key references are made available to the device server is outside the scope of this standard. A device server that supports passing keys by vendor-specific key reference shall include the code for vendor-specific key reference format (see table 147) in the SUPPORTED KEY FORMATS LIST field in the Supported Key Formats page (see 8.5.2.5).

The physical device shall release the resources used to save a set of data encryption parameters (see 4.2.21.8) under the following conditions:

a) the CKOD bit is set to one in the saved data encryption parameters and the volume is demounted;
b) the CKORL bit is set to one and the key scope is set to LOCAL in the saved data encryption parameters and the I_T nexus that established the set of data encryption parameters loses its reservation;
c) the CKORL bit is set to one and the key scope is set to ALL I_T NEXUS in the saved data encryption parameters and the device server experiences a reservation loss (see 3.1.56);
d) the CKORP bit is set to one in the saved data encryption parameters and the device server processes a PERSISTENT RESERVE OUT command with a service action of either PREEMPT or PREEMPT AND ABORT;
e) a microcode update is performed on the device; or
f) a power on condition occurs.

The physical device may release the resources used to save a set of data encryption parameters if:

a) a volume is mounted that does not support data encryption using the algorithm specified by the algorithm index in the data encryption parameter; or
b) other vendor-specific events.

Editors Note 1 - DAP: data encryption parameter is ambiguous to me, need to specify the page/ descriptor.

If a device server processes a Set Data Encryption page with the ENCRYPTION MODE field set to DISABLE and DECRYPTION MODE field set to DISABLE or RAW, the physical device shall:

a) release any resources that it had allocated to store data encryption parameters for the I_T nexus associated with the SECURITY PROTOCOL OUT command and shall change the contents of all memory containing a key value associated with the data encryption parameters that are released; and
b) establish a unit attention condition with the additional sense of DATA ENCRYPTION PARAMETERS CHANGED BY ANOTHER I_T NEXUS for all other I_T nexus that has its registered for encryption unit attentions state set to one (see 4.2.21.7) and is affected by the loss of the key, (i.e., any I_T nexus that is using a data encryption scope of PUBLIC and sharing the keys).

If a device server processes a Set Data Encryption page that includes a key and the SDK bit is set to zero, the device server shall establish a unit attention condition with the additional sense code set to DATA ENCRYPTION PARAMETERS CHANGED BY ANOTHER I_T NEXUS for all other I_T nexus that have their registered for encryption unit attentions state set to one (see 4.2.21.7) and are affected by the change of the key (i.e., any I_T nexus that is using a data encryption scope of PUBLIC and sharing the key), at the physical device shall:

a) release all resources that it had allocated to store a key value set by a previous SECURITY PROTOCOL OUT command from that I_T nexus and shall change the contents of all memory containing a key value associated with the data encryption parameters that are released; and
b) establish a set of data encryption parameters with the values from the Set Data Encryption page.

A physical device shall save at most one set of data encryption parameters with a key scope of ALL I_T NEXUS. If a device server processes a Set Data Encryption page with the SCOPE field set to ALL I_T NEXUS, the device server shall establish a unit attention condition with the additional sense code set to DATA ENCRYPTION PARAMETERS CHANGED BY ANOTHER I_T NEXUS for all other I_T nexus that have their registered for encryption unit attentions state set to one (see 4.2.21.7) and are affected by the change of the key (i.e. any I_T nexus that is using a data encryption scope of PUBLIC and sharing the key) and the physical device shall:

a) release any resources that it had allocated to store data encryption parameters with a key scope value of ALL I_T NEXUS and shall change the contents of all memory containing a key value associated with the data encryption parameters that are released; and
b) establish a set of data encryption parameters with the values from the Set Data Encryption page and a key scope value of ALL I_T NEXUS.

If a vendor-specific event occurs that changes or clears a set of data encryption parameters, the device server shall establish a unit attention condition with the additional sense of DATA ENCRYPTION PARAMETERS CHANGED BY VENDOR SPECIFIC EVENT for any I_T nexus that has its registered for encryption unit attentions state set to one (see 4.2.21.7) and is affected by the change of the key.

4.2.21.7 Saved information per I_T nexus

If the device server supports data encryption it shall save the following information on a per I_T nexus basis:

a) data encryption scope;
b) lock;
c) key instance counter value at lock;
d) key instance counter value assigned to the last key established by a Set Data Encryption page for this I_T nexus with a scope value of LOCAL and the SDK bit is set to zero; and
e) registered for encryption unit attentions state.

The set of possible data encryption scope values for an I_T nexus is:

a) PUBLIC;

b) LOCAL; or
c) ALL I_T NEXUS

If an I_T nexus data encryption scope is set to PUBLIC it indicates the physical device does not have a saved set of data encryption parameters that were established by that I_T nexus. Device servers that support encryption shall support an I_T nexus data encryption scope of PUBLIC.

A device server shall set the data encryption scope for an I_T nexus to LOCAL when it successfully completes the processing of a Set Data Encryption page with a scope of LOCAL from that I_T nexus. The device server shall only use the data encryption parameters established by the Set Data Encryption page with a scope of LOCAL for processing commands from the I_T nexus that established the parameters. A physical device shall revert to using default data encryption parameters for an I_T nexus that is configured with a data encryption scope of LOCAL if the resources used to save the data encryption parameters for the I_T nexus are released.

A device server shall set the data encryption scope for an I_T nexus to ALL I_T NEXUS when it successfully completes the processing of Set Data Encryption page with a scope value of ALL I_T NEXUS from that I_T nexus. At most, one I_T nexus shall be assigned the data encryption scope of ALL I_T NEXUS. If the physical device releases resources used to store a set of data encryption parameters with a key scope of ALL I_T NEXUS, it shall change the data encryption scope for the I_T nexus that established that set of data encryption parameters to PUBLIC. Device servers that support encryption shall support an I_T nexus data encryption scope of ALL I_T NEXUS.

By default, the device server shall set the saved I_T nexus parameters data encryption scope value to PUBLIC and lock value to zero.

The registered for encryption unit attentions state is a single bit state variable that indicates if the device server shall generate unit attention conditions related to encryption status for the I_T nexus. The device server shall set the registered for encryption unit attentions state to one for an I_T nexus if the device server processes a:

a) SECURITY PROTOCOL IN command specifying the Tape Data Encryption protocol from the I_T nexus; or
b) SECURITY PROTOCOL OUT command specifying the Tape Data Encryption protocol from the I_T nexus.

The device server shall set the registered for encryption unit attentions state to zero for an I_T nexus if an I_T nexus loss occurs. The device server shall set the registered for encryption unit attentions state to zero for all I_T nexus if the device server processes a logical unit reset.

4.2.21.8 Data encryption parameters

A device server that supports data encryption shall have the ability to save the following information in the physical device as a set of data encryption parameters when a Set Data Encryption page is processed:

a) for SCSI transport protocols where SCSI initiator device port names are required, the SCSI initiator device port name; otherwise, the SCSI initiator device port identifier;
b) indication of the SCSI target port through which the data encryption parameters were established;
c) key scope;
d) encryption mode;
e) decryption mode;
f) key;
g) supplemental decryption keys where supported;
h) algorithm index;
i) key instance counter;
j) CKOD;
k) CKORL;
l) CKORP;

m) U-KAD;
n) A-KAD;
o) M-KAD;
p) nonce;
q) raw decryption mode disable where supported; and
r) check external encryption mode where supported.

A physical device may have limited resources for storage of sets of data encryption parameters (i.e., it may not have enough resources to store a unique set of data encryption parameters for every I_T nexus that it is capable of managing). A physical device may release a previously established set of data encryption parameters when a Set Data Encryption page is processed and there are no unused resources available. The method of choosing which set of data encryption parameters to release is vendor specific. If the physical device does release a previously established set of data encryption parameters to free the resource, the device server shall establish a unit attention condition for every affected I_T nexus (see 4.2.21.6) that has its registered for encryption unit attentions state set to one (see 4.2.21.7). A physical device is not required to have separate resources to store data encryption parameters for every scope that is supported.

A device server shall support an encryption key scope value of ALL I_T NEXUS and the physical device shall have resources to save one set of data encryption parameters with this scope.

If the device server supports an encryption key scope value of LOCAL, the physical device shall have resources to save one or more sets of data encryption parameters with this scope.

The data encryption parameters that shall be used for an I_T nexus shall be established by the following order of precedence:

a) if the data encryption scope for the I_T nexus is set to LOCAL or ALL I_T NEXUS (see 4.2.21.7), the data encryption parameters set by the last Set Data Encryption page from that I_T nexus; or
b) if the data encryption scope for the I_T nexus is set to PUBLIC:
    1) the data encryption parameters that have been saved by the physical device with a key scope of ALL I_T NEXUS if any data encryption parameters have been saved with this key scope; or
    2) the default data encryption parameters.

4.2.21.9 Data encryption capabilities

A physical device that supports data encryption shall have a set of data encryption capabilities. The set of data encryption capabilities determine the values reported through a SECURITY PROTOCOL IN command specifying the Tape Data Encryption security protocol and the Data Encryption Capabilities page (see clause 8.5.2.4). The set of data encryption capabilities includes the set of data encryption algorithms supported by the physical device.

The set of data encryption capabilities includes some values which may be changed by a method beyond the scope of this standard. The capabilities that may be changed include:

a) the set of data encryption algorithms reported by the device server;
b) encryption capable;
c) decryption capable; and
d) other vendor-specific data encryption capabilities.

4.2.21.10 Key instance counter

The device server shall keep a counter for each key that it is managing called the key instance counter. All key instance counters shall be set to zero when a power on condition occurs. Any other event that sets, clears, or changes a parameter in a set of data encryption parameters, except the supplemental decryption keys, shall cause the key instance counter for that set of data encryption parameters to be incremented. The value of the key

instance counter associated with the currently selected key for an I_T nexus is reported in the Data Encryption Status page of the SECURITY PROTOCOL IN command. The key instance counters are 32 bits and shall roll over to zero when incremented past their maximum value.

4.2.21.11 Encryption mode locking

There are conditions outside of the control of an application client which cause the physical device to release the resources used to save the data encryption parameters (see 4.2.21.6) or change the data encryption parameters used to control the encryption of logical blocks. Each of these conditions cause the device server to establish a unit attention condition to report the change of operating mode, but the unit attention condition may not always be reported to the application client through protocol bridges and driver stacks.

The LOCK bit in the Set Data Encryption page is set to one to lock the I_T nexus that issued the SECURITY PROTOCOL OUT command to the set of data encryption parameters established at the completion of the processing of the command. The I_T nexus remains locked to that set of data encryption parameters and key instance counter value until a hard reset condition occurs or another SECURITY PROTOCOL OUT command including a Set Data Encryption page from the same I_T nexus is processed.

If the device server processes a WRITE(6) or WRITE(16) command for an I_T nexus that is locked to a set of data encryption parameters and key instance counter, and the key instance counter value has changed since the time it was locked, the device server shall terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to DATA ENCRYPTION KEY INSTANCE COUNTER HAS CHANGED. All subsequent WRITE(6) and WRITE(16) commands shall also be terminated in this manner until a hard reset condition occurs or a SECURITY PROTOCOL OUT command including a Set Data Encryption page from the same I_T nexus is processed.

Editors Note 2 - DAP: Review all (red) text in this subclause with respect to data vs logical block and I_T nexus vs I_T_L nexus.

4.2.21.12 Nonce generation

For a given encryption algorithm, the physical device may:

a) not require a nonce value;
b) generate its own nonce value;
c) require a nonce value or part of the nonce value be provided by the application client; or
d) be configurable with respect to the source of the nonce value.

The device server reports its capability with respect to nonce values in the Data Encryption Algorithm descriptor(s) (see 8.5.2.4). If the device server reports that it requires a nonce value from the application client and a Set Data Encryption page is processed that does not include a nonce value descriptor, the device server shall terminate the command with CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INCOMPLETE KEY-ASSOCIATED DATA SET.

4.2.21.13 Unauthenticated key-associated data (U-KAD) and authenticated key-associated data (A-KAD)

Some encryption algorithms allow or require the use of additional data which is associated with the key and the plaintext, but which is not encrypted. It may be authenticated by being included in the message authentication code (MAC) calculations for the encrypted plaintext if such a MAC exists, or unauthenticated by not being included in these calculations.

NOTE 12 A key-identifier or key reference may be stored in the U-KAD or A-KAD.

The U-KAD field is provided for applications that do not require the key-associated data to be protected by an MAC.

4.2.21.14 Metadata key-associated data (M-KAD)

Some encryption algorithms allow or require the use of additional data which is associated with the key and every logical block encrypted with that key. This data is contained in an M-KAD descriptor.

4.2.22 External data encryption control

4.2.22.1 External data encryption control overview

A physical device that supports data encryption may support external data encryption control and provide the ability for an external entity to configure data encryption capabilities or data encryption parameters using an external interface not specified by this standard (e.g., an ADC device server or a management interface).

4.2.22.2 External data encryption control of data encryption capabilities

4.2.22.2.1 External data encryption control of data encryption capabilities overview

If the physical device has a saved set of data encryption parameters associated with this device server or has a medium mounted, then the physical device shall not allow external data encryption control of data encryption capabilities. If the physical device does not have a set of data encryption parameters associated with this device server and does not have a medium mounted, then external data encryption control may be used to change the data encryption capabilities.

If external data encryption control is used to change any of the data encryption capabilities of the physical device, then the device server shall establish a unit attention condition with the additional sense code of DATA ENCRYPTION CAPABILITIES CHANGED for all I_T nexus that have their registered for encryption unit attentions state set to one (see 4.2.21.7).

4.2.22.2.2 External data encryption control of encryption algorithm support

External data encryption control may be used to change the device server encryption algorithm support by configuring the physical device to:

a) disable a supported data encryption algorithm; or
b) prevent device server control of data encryption parameters.

If a supported encryption algorithm has been disabled then:

a) the physical device shall not accept data encryption parameters specifying that algorithm; and
b) the device server shall:
   A) not report the disabled data encryption algorithm in the Data Encryption Capabilities page; or
   B) report the encryption algorithm in the Data Encryption Capabilities page with the DISABLED bit set to one.

If external data encryption control has been used to configure the physical device to prevent device server control of data encryption parameters, then the device server shall:

a) terminate a SECURITY PROTOCOL OUT command that attempts to establish or clear a set of data encryption parameters with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to DATA ENCRYPTION CONFIGURATION PREVENTED; and

b) set the CFG_C (see 8.5.2.4) field in the Data Encryption Capabilities page to 10b (i.e., the physical device is configured to not allow this device server to establish or change data encryption parameters) and:
   A) not report any encryption algorithms in the Data Encryption Capabilities page; or
   B) report all of the supported data encryption algorithms in the Data Encryption Capabilities page with the DECRYPT_C field set to capable with external control and the ENCRYPT_C field set to capable with external control.

NOTE 13 The SECURITY PROTOCOL IN command specifying the Tape Data Encryption security protocol and the Data Encryption Status page may be used to determine whether external data encryption control has been used to provide a set of data encryption parameters.

4.2.22.3 External data encryption control of data encryption parameters

4.2.22.3.1 External data encryption control of data encryption parameters overview

External data encryption control may be used to control data encryption parameters by using:

1) a data encryption parameters request policy to set a data encryption parameters request indicator to TRUE;
2) a data encryption parameters period to determine how long to wait for the data encryption parameters request indicator to be set to FALSE; and
3) the set of data encryption parameters that have been set in the physical device.

A physical device that supports external data encryption control shall contain a data encryption parameters request policy (see 4.2.22.3.2) and a set of data encryption parameters request indicators (see 4.2.22.3.3).

4.2.22.3.2 Data encryption parameters request policy

The data encryption parameters request policy determines when the physical device shall request a set of data encryption parameters from an entity using external data encryption control. The data encryption parameters request policy shall contain a data encryption parameters for encryption request policy and a data encryption parameters for decryption request policy.

External data encryption control sets the data encryption parameters for encryption parameters request policy (see table 15) and the data encryption parameters for decryption request policy (see table 16) to indicate to the physical device what events shall cause a data encryption parameters request indicator to be set to TRUE (see 4.2.22.3.3). If external data encryption control is not being used, then the data encryption control policies shall be set to defaults.

The data encryption parameters for encryption request policies setting are specified in table 15.

Table 15 — Data encryption parameters for encryption request policies

Policy	Description
No data encryption parameter requests	The physical device shall not set the data encryption parameters for encryption request indicator to TRUE. This is the default setting for the data encryption parameters for encryption request policy.

Request data encryption parameters every reposition	The physical device shall set the data encryption parameters for encryption request indicator to TRUE when the device server processes the first:
	a) WRITE(6) command;
	b) WRITE(16) command;
	c) WRITE FILEMARKS(6)[a] command with a non-zero FILEMARK COUNT field; or
	d) WRITE FILEMARKS(16)[a] command with a non-zero FILEMARK COUNT field;
	after:
	a) an ERASE(6) command;
	b) an ERASE(16) command;
	c) a FORMAT MEDIUM command;
	d) a LOCATE(10) command;
	e) a LOCATE(16) command;
	f) a LOAD UNLOAD command;
	g) a REWIND command;
	h) a READ(6) command;
	i) a READ(16) command;
	j) a READ REVERSE(6) command;
	k) a READ REVERSE(16) command;
	l) a VERIFY(6) command;
	m) a VERIFY(16) command;
	n) a SPACE(6) command; or
	o) a SPACE(16) command.

Request data encryption parameters when not set	The physical device shall set the data encryption parameters for encryption request indicator to TRUE before accepting any data into the buffer or adding any filemarks to the buffer if running in buffered mode or to the medium if running in unbuffered, when the device server processes the first:
	a) WRITE(6) command;
	b) WRITE(16) command;
	c) WRITE FILEMARKS(6)[a] command with a non-zero FILEMARK COUNT field; or
	d) WRITE FILEMARKS(16)[a] command with a non-zero FILEMARK COUNT field;
	after:
	a) there is not an established set of data encryption parameters; or
	b) an event that causes the data decryption parameters request indicator to be set to TRUE.

a) The WRITE FILEMARKS command is included in the list of commands that cause the data encryption parameters for encryption request indicator to be set to TRUE to prevent an application client from writing a filemark as part of a new operation (e.g., a backup operation starting with a WRITE FILEMARKS command and followed by a series of WRITE commands) when the operation will not be successful due to a failure to retrieve a set of data encryption parameters.

The data encryption parameters for decryption request policies setting are specified in table 16.

Table 16 — Data encryption parameters for decryption request policies

Policy	Description
No data decryption parameter requests	The physical device shall not set the data encryption parameters for decryption request indicator to TRUE. This is the default setting for the data encryption parameters for decryption request policy.

Request data decryption parameters as needed	The physical device shall set the data encryption parameters for decryption request indicator to TRUE when the physical device detects that the current set of data encryptionparameters is not correct for a logical block being processed as a result of processing a:
	a) a READ(6) command;
	b) a READ(16) command;
	c) a READ REVERSE(6) command;
	d) a READ REVERSE(16) command;
	e) a VERIFY(6) command with the BYTCMP bit set to one; or
	f) a VERIFY(16) command with the BYTCMP bit set to one.

The data encryption parameters for encryption request policy and the data encryption parameters for decryption request policy settings shall be set to defaults upon:

a) a hard reset condition; or
b) other vendor specific events.

4.2.22.3.3 Data encryption parameters request indicators

The data encryption parameters request indicators indicate when the physical device requires a set of data encryption parameters from a entity using external data encryption control. The data encryption parameters request indicators shall contain a data encryption parameters for encryption request indicator and a data encryption parameters for decryption request indicator.

The data encryption parameters for encryption request indicator settings are specified in table 17.

Table 17 — Data encryption parameters for encryption request indicator settings

Setting	Description
TRUE	The physical device is waiting for the data encryption parameters for encryption request indicator to be set to FALSE before continuing to process the task in the enabled task state (e.g., an ADC device server processes a SECURITY PROTOCOL OUT command with a DATA ENCRYPTION PARAMETERS COMPLETE page and the clear encryption parameters request (CEPR) bit set to one, see ADC-3).
FALSE	The physical device is not waiting for the data encryption parameters for encryption request indicator to be set to FALSE before continuing to process the task in the enabled task state. This is the default setting for the data encryption parameters for encryption request indicator.

When the data encryption parameters for encryption request indicator is set to FALSE, then the device server shall resume processing of the command that caused the data encryption parameters for encryption request indicator to be set to TRUE.

The data encryption parameters for decryption request indicator settings are specified in table 18.

Table 18 — Data encryption parameters for decryption request indicator settings

Setting	Description
TRUE	The physical device is waiting for the data encryption parameters for decryption request indicator to be set to FALSE before continuing to process the task in the enabled task state (e.g., an ADC device server processes a SECURITY PROTOCOL OUT command with a DATA ENCRYPTION PARAMETERS COMPLETE page and the clear encryption parameters request (CEPR) bit set to one, see ADC-3).
FALSE	The physical device is not waiting for the data encryption parameters for decryption request indicator to be set to FALSE before continuing to process the task in the enabled task state. This is the default setting for the data encryption parameters for decryption request indicator.

The physical device shall not change the logical position while the data encryption parameters for decryption request indicator is set to TRUE.

When the data encryption parameters for decryption request indicator is set to FALSE, then the device server shall resume processing of the command that caused the data encryption parameters for decryption request indicator to be set to TRUE.

The data encryption parameters for encryption request indicator and the data encryption parameters for decryption request indicator shall be set to defaults on:

a) a hard reset condition;
b) a volume is demounted;
c) a data encryption parameters request period timeout (see 4.2.22.3.4); or
d) successfully processing a task management request that terminates processing of the task.

When the data encryption parameters for decryption request indicator is set to FALSE or the data encryption parameters for encryption request indicator is set to FALSE, then the data encryption period timer shall be set to zero.

4.2.22.3.4 Data encryption parameters period settings

The data encryption parameters period settings contain values that:

a) determine how long the physical device waits for a set of data encryption parameters;
b) track how long the physical device has waited for a set of data encryption parameters after a data encryption parameters request indicator is set to TRUE; and
c) indicate when the time to wait for a set of data encryption parameters period has expired.

The data encryption parameters period settings shall contain a data encryption parameters period time, a data encryption period timer, and a data encryption parameters period expired indicator.

The data encryption parameters period time shall contain a value indicating the amount of time that the physical device shall wait for a set of data encryption parameters if:

a) the data encryption parameters for encryption request indicator (see 4.2.22.3.3) is set to TRUE; or
b) the data encryption parameters for decryption request indicator (see 4.2.22.3.3) is set to TRUE.

The data encryption period timer shall contain the time since:

a) the data encryption parameters for encryption request indicator was set to TRUE; or
b) the data encryption parameters for decryption request indicator was set to TRUE.

The data encryption period timer shall be set to zero when:

a) the data encryption parameters for encryption request indicator is set to FALSE; or
b) the data encryption parameters for decryption request indicator is set to FALSE.

The values of the data encryption period timer expired indicator are show in table 19.

Table 19 — Data encryption period timer expired indicator

Setting	Description
TRUE	The data encryption period timer has expired.
FALSE	The data encryption period timer has not expired.

If the data encryption period timer reaches the data encryption period time, then the:

a) data encryption period timer expired shall be set to TRUE;
b) data encryption parameters for encryption request indicator shall be set to FALSE;
c) data encryption parameters for decryption request indicator shall be set to FALSE; and
d) the device server shall terminate the command that caused a request indicator to be set to TRUE with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to EXTERNAL DATA ENCRYPTION CONTROL TIMEOUT.

4.2.22.4 Exclusive control of data encryption parameters by external data encryption control

An entity outside the scope of this standard may configure the physical device for exclusive control of data encryption using external data encryption control. If external data encryption control is used to configure the physical device to prevent control of the data encryption parameters by this device server, then the device server shall respond to a SECURITY PROTOCOL IN command specifying the Tape Data Encryption security protocol and the Data Encryption Status page with the PARAMETERS CONTROL field set to 011b or 100b.

4.2.22.5 External data encryption control error conditions

If external data encryption control is being used to control the data encryption parameters and the external data encryption control data encryption parameters lookup process returns an error, then the device server shall terminate the command that initiated the data encryption parameters lookup process with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to the value of the external data encryption control additional sense code in the physical device, or to EXTERNAL DATA ENCRYPTION CONTROL ERROR if the external data encryption control additional sense code is set to NO ADDITIONAL SENSE INFORMATION.

An application client may use the DTD Status log page to get information about the error that occurred (see ADC-3).

4.2.23 Data encryption key protection

4.2.23.1 Data encryption key protection overview

A device server that supports data encryption should protect data encryption keys from disclosure. Key disclosure may be mitigated by several countermeasures such as key wrapping and/or securing the channel used to transmit the key.

4.2.23.2 Encryption key protection using security associations

Security associations (see SPC-4) may be used to protect data encryption keys and associated data encryption parameters from disclosure and modification. A device server that supports SAs as a way to protect keys may require that all key operations be protected using an SA.

NOTE 14 NIST SP800-57 Part 1 discourages combining non-comparable strength algorithms.

4.2.23.3 Key wrapping using public key cryptography

A device server that supports public key cryptography for key wrapping, shall have a secret private key and a public key. The public key is used for wrapping key material sent to the device server. The private key is used by the device server to unwrap the keys sent to it. The entity wrapping the encryption key is assured that only the device server that owns the private portion of this public key can unwrap the encryption key.

A device server that supports public key cryptography for key wrapping may ensure the authenticity of the wrapped key by verifying the key wrapper's signature. The key wrapping entity's secret private key is used to sign the wrapped key. The key signing entity's public key is used by the device server to verify the signature.

A device server that supports signature verification shall store the key wrappers' public keys in an authorization white list. While these public keys are not secret, the device server shall maintain the authorization white list in a way that will prevent an attacker from modifying a public key or even injecting his own (such operations will grant the attacker the ability to send wrapped keys to the device server).

A device server that supports signature verification should be able to store a minimum of four public keys for signature verification to allow for two key wrapping entities and the ability to replace these keys.

The method of adding signature verification public keys to the authorization white list is beyond the scope of this standard.

Editors Note 3 - DAP: Add a definition for authorization white list.

4.2.24 Appending data to a volume containing encrypted data

A volume contains no encrypted logical blocks, all encrypted logical blocks, or a mixture of encrypted logical blocks and unencrypted logical blocks.

A device server that supports encryption should be capable of determining when a mounted volume contains an encrypted logical block. The device server reports its capability of determining if a volume contains an encrypted logical block using the VCED_C bit in the Data Encryption Algorithm descriptor (see 8.5.2.4). If the device server is capable of determining whether a mounted volume contains an encrypted logical block, it should support a value of one in the VCEDRE bit of the Device Configuration Extension mode page (see 8.3.8).

The device server shall terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to ENCRYPTION PARAMETERS NOT USEABLE if the following are true:

a) the VCEDRE bit in the Device Configuration Extension mode page (see 8.3.8) is set to one;
b) the vced bit in the Data Encryption status page (see 8.5.2.7) is set to one;
c) the encryption mode of the set of data encryption parameters in use by the I_T nexus on which the command arrived is set to DISABLE;
d) the logical object identifier does not equal zero; and
e) the command is a:
   A) WRITE(6);
   B) WRITE(16);
   C) WRITE FILEMARKS(6); or
   D) WRITE FILEMARKS(16).

4.2.25 Self-test operations

The logical position following the completion of a self-test is not specified by this standard. See SPC-4.

4.2.26 Capability-based command security

4.2.26.1 Capability-based command security (CbCS) overview

CbCS is a credential-based system that manages access to a logical unit or a volume. See SPC-4.

4.2.26.2 Association between commands and permission bits

Table 20 specifies the permissions required in the PERMISSIONS BIT MASK field in the capability descriptor of a CbCS extension descriptor (see SPC-4) for each SCSI command specified in this standard. The permissions listed in table 20 are specified in SPC-4. This standard does not define any permissions specific to the sequential-access device type.

Table 20 — Association between commands and CbCS permissions

Requested Command	Permissions				
	DATA READ	DATA WRITE	PARM READ	PARM WRITE	PHY ACC
ERASE(6)		V			
ERASE(16)		V			
FORMAT MEDIUM		V		V	
LOAD UNLOAD					V
LOCATE(10)	V				
LOCATE(16)	V				
PREVENT ALLOW MEDIUM REMOVAL					V
READ(6)	V				
READ(16)	V				

Key: V = The secure CDB processor shalll process the requested command only if the corresponding bit is set in the PERMISSIONS BIT MASK field in the capability descriptor of a CbCS extension descriptor.

Table 20 — Association between commands and CbCS permissions (Continued)

Requested Command	Permissions				
	DATA READ	DATA WRITE	PARM READ	PARM WRITE	PHY ACC
READ BLOCK LIMITS			V		
READ POSITION	V				
READ REVERSE(6)	V				
READ REVERSE(16)	V				
RECOVER BUFFERED DATA	V	V			
REPORT DENSITY SUPPORT			V		
REWIND	V				
SET CAPACITY		V		V	
SPACE(6)	V				
SPACE(16)	V				
VERIFY(6)	V				
VERIFY(16)	V				
WRITE(6)		V			
WRITE(16)		V			
WRITE FILEMARKS(6)		V			
WRITE FILEMARKS(16)		V			

Key: V = The secure CDB processor shalll process the requested command only if the corresponding bit is set in the PERMISSIONS BIT MASK field in the capability descriptor of a CbCS extension descriptor.

5 Explicit address command descriptions for sequential-access devices

5.1 Summary of commands for explicit address mode

The explicit address command set for sequential-access devices shall be as shown in table 21. Commands specified as mandatory in table 21 shall be implemented if the explicit address command set is supported.

Refer to table 14 for a description of reservations.

The following command codes are vendor specific: 02h, 06h, 07h, 09h, 0Ch, 0Dh, and 0Eh.

Table 21 — Explicit address command set for sequential-access devices

Command Name	Type	OpCode	Synchronize Operation Required[a]	Command Type	Reference
ACCESS CONTROL IN	O	86h	No	G	SPC-4
ACCESS CONTROL OUT	O	87h	No	G	SPC-4
CHANGE ALIAS	O	A4h/0Bh[c]	No	G	SPC-4
ERASE(16)	M	93h	Yes	W-E	5.2
EXTENDED COPY	O	83h	No	W or R[b]	SPC-4
FORMAT MEDIUM	O	04h	No	W	7.1
INQUIRY	M	12h	No	G-A	SPC-4
LOAD UNLOAD	O	1Bh	Yes	G	7.2
LOCATE(16)	M	92h	Yes	G-E	7.3
LOG SELECT	O	4Ch	No	G	SPC-4
LOG SENSE	O	4Dh	No	G-A	SPC-4
MODE SELECT(6)	M	15h	Yes[a]	W or R[b]	SPC-4
MODE SELECT(10)	O	55h	Yes[a]	W or R[b]	SPC-4
MODE SENSE(6)	M	1Ah	No	G	SPC-4

Key: M = Command implementation is mandatory.
O = Command implementation is optional.
X = See referenced standard or subclause.
Z = Command implementation is specified in a previous standard.
R = Read type command.
W = Write type command.
G = Generic type command.
E = Explicit command.
A = Allowed command while in write capable state.

a. Refer to 4.2.10.
b. This command has some specific actions that fall under write type commands and some that fall into read type commands.
c. This command is defined by a combination of operation code and service action. The operation code value is shown preceding the slash and the service action value is shown after the slash.

Table 21 — Explicit address command set for sequential-access devices (Continued)

Command Name	Type	OpCode	Synchronize Operation Required[a]	Command Type	Reference
MODE SENSE(10)	O	5Ah	No	G	SPC-4
PERSISTENT RESERVE IN	M	5Eh	No	G	SPC-4
PERSISTENT RESERVE OUT	M	5Fh	No	G	SPC-4
PREVENT ALLOW MEDIUM REMOVAL	O	1Eh	No	G-A	SPC-4
READ(16)	M	88h	Yes	R-E	5.3
READ ATTRIBUTE	O	8Ch	No	G	SPC-4
READ BLOCK LIMITS	M	05h	No	G-A	7.5
READ BUFFER	O	3Ch	Yes	G	SPC-4
READ POSITION	M	34h	No	G-A	7.6
READ REVERSE(16)	O	81h	Yes	R-E	5.4
RECEIVE COPY RESULTS	O	84h	No	G	SPC-4
RECEIVE DIAGNOSTIC RESULTS	O	1Ch	No	G	SPC-4
RECOVER BUFFERED DATA	O	14h	May	R	7.7
REPORT ALIAS	O	A3h/0Bh[c]	No	G	SPC-4
REPORT DENSITY SUPPORT	M	44h	No	G-A	7.8
REPORT DEVICE IDENTIFIER	O	A3h/05h[c]	No	G	SPC-4
REPORT LUNS	X	A0h	No	G-A	SPC-4
REPORT SUPPORTED OPERATION CODES	O	A3h/0Ch[c]	No	G	SPC-4
REQUEST SENSE	M	03h	No	G	SPC-4
REPORT TARGET PORT GROUPS	O	A3h/0Ah[c]	No	G	SPC-4
REWIND	M	01h	Yes	G	7.9
SECURITY PROTOCOL IN	O	A2h	No	G	SPC-4
SECURITY PROTOCOL OUT	O	B5h	No	G	SPC-4

Key: M = Command implementation is mandatory.
O = Command implementation is optional.
X = See referenced standard or subclause.
Z = Command implementation is specified in a previous standard.
R = Read type command.
W = Write type command.
G = Generic type command.
E = Explicit command.
A = Allowed command while in write capable state.

a. Refer to 4.2.10.
b. This command has some specific actions that fall under write type commands and some that fall into read type commands.
c. This command is defined by a combination of operation code and service action. The operation code value is shown preceding the slash and the service action value is shown after the slash.

Table 21 — Explicit address command set for sequential-access devices (Continued)

Command Name	Type	OpCode	Synchronize Operation Required[a]	Command Type	Reference
SEND DIAGNOSTIC	M	1Dh	Yes[a]	W or R[b]	SPC-4
SET CAPACITY	O	0Bh	May	W	7.10
SET DEVICE IDENTIFIER	O	A4h/06h[c]	No	G	SPC-4
SET TARGET PORT GROUPS	O	A4h/0Ah[c]	No	G	SPC-4
SPACE(16)	O	91h	May	G-E	7.11
TEST UNIT READY	M	00h	No	G	SPC-4
VERIFY(16)	O	8Fh	Yes	R-E	5.5
WRITE(16)	M	8Ah	No	W-E	5.6
WRITE ATTRIBUTE	O	8Dh	No	G	SPC-4
WRITE BUFFER	O	3Bh	Yes[a]	G	SPC-4
WRITE FILEMARKS(16)	M	80h	May	W-E	5.7
Obsolete	Z	A7h			
Obsolete	Z	B4h			

Key: M = Command implementation is mandatory.
O = Command implementation is optional.
X = See referenced standard or subclause.
Z = Command implementation is specified in a previous standard.
R = Read type command.
W = Write type command.
G = Generic type command.
E = Explicit command.
A = Allowed command while in write capable state.

a. Refer to 4.2.10.
b. This command has some specific actions that fall under write type commands and some that fall into read type commands.
c. This command is defined by a combination of operation code and service action. The operation code value is shown preceding the slash and the service action value is shown after the slash.

5.2 ERASE(16) command

The ERASE(16) command (see table 22) causes part or all of the medium to be erased beginning at the logical object identifier and partition specified in the command descriptor block. Prior to performing the erase operation, the device server shall perform a synchronize operation (see 4.2.10).

Table 22 — ERASE(16) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (93h)							
1	Reserved				FCS	LCS	IMMED	LONG
2	Reserved		METHOD		Reserved		SMD	VCM
3	PARTITION							
4	(MSB)							
5								
6								
7								
8				LOGICAL OBJECT IDENTIFIER				
9								
10								
11								(LSB)
12	Reserved							
13	Reserved							
14	Reserved							
15	CONTROL							

A first command in sequence (FCS) bit of one specifies this command is the first command in a tagged write sequence. An FCS bit of zero specifies this command is not the first command in a tagged write sequence.

A last command in sequence (LCS) bit of one specifies this command is the last command in a tagged write sequence. An LCS bit of zero specifies this command is not the last command in a tagged write sequence.

An immediate (IMMED) bit of zero specifies the device server shall not return status until the erase operation has completed. Interpretation of an IMMED bit of one depends on the value of the LONG bit, see below. However, for all values of the LONG bit, if CHECK CONDITION status is returned for an ERASE(16) command with an IMMED bit of one, the erase operation shall not be performed.

NOTE 15 Application clients should set the IMMED bit set to zero to guarantee the operation has completed successfully when setting the METHOD field to 10b. When the METHOD field is set to 10b, the duration of the processing may be extended (i.e., may be longer than just setting the LONG bit to one).

A LONG bit of one specifies all remaining medium shall be erased beginning at the specified logical object identifier and partition shall be erased using the method specified by the METHOD field value (see table 23). If the format on the medium specifies a recorded indication of EOD (see 3.1.18), the erase operation shall establish an EOD indication at the specified location as part of the erase operation. If the IMMED bit is one, the device server shall return status as soon as all buffered logical objects have been written to the medium and the command descriptor

block of the ERASE(16) command has been validated. The logical position following an ERASE(16) command with a LONG bit of one is not specified by this standard.

NOTE 16 Some logical units may reject an ERASE(16) command if the logical object identifier is not zero.

A LONG bit of zero specifies the device server shall perform the action specified by the short erase mode field in the Device Configuration Extension mode page (see 8.3.8) at the logical object identifier and partition specified in the command. The logical position following a ERASE(16) command with a LONG bit of zero shall be at the specified logical object identifier and partition. If the IMMED bit is one, the device server shall return status as soon as the command descriptor block has been validated.

The METHOD field specifies the erase method that shall be used to erase data. Table 23 defines the METHOD field values. If the LONG bit is set to zero, the METHOD field only applies to data outside the user data area(s).

Table 23 — METHOD field values

Value	Description
00b	Vendor specific
01b	The device server shall erase or over-write the volume with a format-specific pattern. Upon successful procesiing of the command, the volume may contain fragments of data specified for erasure. The data specified for erasure shall not be recognizable as valid user data using normal volume processing methods.
10b	The device server shall erase or over-write the volume with a format-specific pattern(s). Upon successful processing of the command, the volume shall not contain fragments of data specified for erasure.
11b	Reserved

NOTE 17 The METHOD field set to a value of 10b is intended to support data shredding (e.g., sanitization as specified in DoD 5220.22-M).

If the Security Meta-Data (SMD) bit is set to one, the device server shall alter the security meta-data stored on the volume with the method specified by the METHOD field. If the SMD bit is set to zero, the device server handling of the Security Meta-Data stored on the volume is vendor specific.

If the Vendor-specific Control Meta-data (VCM) bit is set to one, the device server shall alter the vendor-specific control meta-data stored on the volume with the method specified by the METHOD field. If the VCM bit is set to zero, the device server handling of the vendor-specific control meta-data stored on the volume is vendor specific.

If the logical unit encounters early-warning during an ERASE(16) command, and any buffered logical objects remain to be written, the device server action shall be as defined for the early-warning condition of the WRITE(16) command (see 5.6). If the LONG bit is zero, the erase operation shall terminate with CHECK CONDITION status and the sense data shall be set as defined for the WRITE(16) command. Any count of pending buffered erases shall not be reported as part of the value returned in the INFORMATION field or in the READ POSITION response data.

The PARTITION and LOGICAL OBJECT IDENTIFIER fields specify the position at which the ERASE(16) command shall start. If the current position does not match the specified LOGICAL OBJECT IDENTIFIER and PARTITION fields, the device server shall perform a locate operation to the specified logical object identifier and partition prior to performing the erase operation. If the locate operation fails, the device server shall return CHECK CONDITION status and the additional sense code shall be set to LOCATE OPERATION FAILURE. The logical position is undefined following a locate operation failure with a LONG bit of zero.

See 4.2.5 for a description of the effect of a PEWZ on the completion of the ERASE(16) command.

5.3 READ(16) command

The READ(16) command (see table 24) requests that the device server transfer one or more logical block(s) to the application client beginning at the logical object identifier and partition specified in the command descriptor block. Prior to performing the read operation, the device server shall perform a synchronize operation (see 4.2.10).

Table 24 — READ(16) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (88h)							
1	Reserved						SILI	FIXED
2	Reserved							
3	PARTITION							
4	(MSB)							
5								
6								
7								
8				LOGICAL OBJECT IDENTIFIER				
9								
10								
11								(LSB)
12	(MSB)							
13				TRANSFER LENGTH				
14								(LSB)
15	CONTROL							

The FIXED bit specifies whether fixed-block transfers or variable-block transfers are to be used. Refer to the READ BLOCK LIMITS command (see 7.5) for additional information about fixed-block transfers and variable-block transfers.

If the FIXED bit is one, the TRANSFER LENGTH specifies the number of fixed-length blocks to be transferred, using the current block length reported in the mode parameters block descriptor (see SPC-4). If the FIXED bit is zero, a variable-length block is requested with the TRANSFER LENGTH specifying the maximum number of bytes allocated for the returned logical block.

A successful READ(16) command with a FIXED bit of one shall transfer the requested transfer length times the current block length in bytes to the application client. A successful READ(16) command with a FIXED bit of zero shall transfer the requested transfer length in bytes to the application client. Upon completion, the logical position shall be after the last logical block transferred (end-of-partition side).

If the suppress incorrect-length indicator (SILI) bit is one and the FIXED bit is zero, the device server shall:

a) report CHECK CONDITION status for an incorrect-length condition only if the overlength condition exists and the BLOCK LENGTH field in the mode parameter block descriptor is nonzero (see SPC-4); or
b) not report CHECK CONDITION status if the only error is the underlength condition, or if the only error is the overlength condition and the BLOCK LENGTH field of the mode parameters block descriptor is zero.

NOTE 18 Since the residual information normally provided in the INFORMATION field of the sense data may not be available when the SILI bit is set to one, other methods for determining the actual block length should be used (e.g., including length information in the logical block).

If the SILI bit is one and the FIXED bit is one, the device server shall terminate the command with CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST with an additional sense code of INVALID FIELD IN CDB.

If the SILI bit is zero and an incorrect-length logical block is read, CHECK CONDITION status shall be returned and the ILI and VALID bits shall be set to one in the sense data with an additional sense code of NO ADDITIONAL SENSE INFORMATION. Upon termination, the logical position shall be after the incorrect-length logical block (end-of-partition side). If the FIXED bit is one, the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks read (not including the incorrect-length logical block). If the FIXED bit is zero, the INFORMATION field shall be set to the requested transfer length minus the actual logical block length. Logical units that do not support negative values shall set the INFORMATION field to zero if the overlength condition exists.

NOTE 19 In the above case with the FIXED bit of one, only the position of the incorrect-length logical block may be determined from the sense data. The actual length of the incorrect logical block is not reported. Other means may be used to determine its actual length (e.g., read it again with the fixed bit set to zero).

The LOGICAL OBJECT IDENTIFIER and PARTITION fields specify the position at which the READ(16) command shall start. If the TRANSFER LENGTH field is not set to zero and the current logical position does not match the specified LOGICAL OBJECT IDENTIFIER and PARTITION fields, the device server shall perform a locate operation to the specified logical object identifier and partition prior to performing the read operation. If the locate operation fails, the device server shall return CHECK CONDITION status and the additional sense code shall be set to LOCATE OPERATION FAILURE. The INFORMATION field in the sense data shall be set to the requested transfer length. Following a locate operation failure the logical position is undefined.

If the TRANSFER LENGTH field is set to zero, no data shall be transferred and the current logical position shall not be changed. This condition shall not be considered an error.

In the case of an unrecovered read error, if the FIXED bit is one, the sense data VALID bit shall be set to one and the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks read (not including the unrecovered logical blocks). If the FIXED bit is zero, the sense data VALID bit shall be set to one and the INFORMATION field shall be set to the requested transfer length. Upon termination, the logical position shall be after the unrecovered logical block.

If the device server encounters a filemark during a READ(16) command, CHECK CONDITION status shall be returned and the FILEMARK and VALID bits shall be set to one in the sense data. The sense key shall be set to NO SENSE or RECOVERED ERROR, as appropriate, and the additional sense code shall be set to FILEMARK DETECTED. Upon termination, the logical position shall be after the filemark (end-of-partition side). If the FIXED bit is one, the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks read. If the FIXED bit is zero, the INFORMATION field shall be set to the requested transfer length.

If the device server encounters early-warning during a READ(16) command and the REW bit is set to one in the Device Configuration mode page (see 8.3.3), CHECK CONDITION status shall be returned upon completion of the current logical block. The sense key shall be set to NO SENSE or RECOVERED ERROR, as appropriate, and the additional sense code shall be set to END-OF-PARTITON/MEDIUM DETECTED. The EOM and VALID bits shall be

set to one in the sense data. Upon termination, the logical position shall be after the last logical block transferred (end-of-partition side). If the FIXED bit is one, the INFORMATION field in the sense data shall be set to the requested transfer length minus the actual number of logical blocks read. If the FIXED bit is zero, the INFORMATION field in the sense data shall be set to the requested transfer length minus the actual logical block length. The device server shall not return CHECK CONDITION status when early-warning is encountered if the REW bit is zero.

NOTE 20 A REW bit of one is not recommended for most applications since read data may be present after early-warning.

If the device server encounters end-of-data during a READ(16) command, CHECK CONDITION status shall be returned, the sense key shall be set to BLANK CHECK, and the VALID bit shall be set to one in the sense data. If end-of-data is encountered at or after early-warning, the EOM bit shall also be set to one. Upon termination, the logical position shall be immediately after the last recorded logical object (end-of-partition side). If the FIXED bit is one, the INFORMATION field in the sense data shall be set to the requested transfer length minus the actual number of logical blocks read. If the FIXED bit is zero, the INFORMATION field shall be set to the requested transfer length.

If the device server encounters end-of-partition during a READ(16) command, CHECK CONDITION status shall be returned, the sense key shall be set to MEDIUM ERROR, and the EOM and VALID bits shall be set to one in the sense data. The medium position following this condition is not defined. If the FIXED bit is one, the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks read. If the FIXED bit is zero, the INFORMATION field in the sense data shall be set to the requested transfer length.

NOTE 21 If a READ(16) command terminates with an error condition other than ILLEGAL REQUEST, and no data transfer has occurred, the logical position of the medium is undefined. The application client should issue a READ POSITION(16) command to determine the logical position.

5.4 READ REVERSE(16) command

The READ REVERSE(16) command (see table 25) requests that the device server transfer one or more logical block(s) to the application client beginning at the logical object identifier and partition specified in the command descriptor block. Prior to performing the read reverse operation, the device server shall perform a synchronize operation (see 4.2.10).

Table 25 — READ REVERSE(16) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (81h)							
1	Reserved					BYTORD	SILI	FIXED
2	Reserved							
3	PARTITION							
4	(MSB)							
5								
6								
7								
8				LOGICAL OBJECT IDENTIFIER				
9								
10								
11								(LSB)
12	(MSB)							
13				TRANSFER LENGTH				
14								(LSB)
15	CONTROL							

This command is similar to the READ(16) command except that medium motion is in the reverse direction. Upon completion of a READ REVERSE(16) command, the logical position shall be before the last logical block transferred (beginning-of-partition side).

A byte order (BYTORD) bit of zero specifies all logical block(s), and the byte(s) within the logical block(s), are transferred in the reverse order. The order of bits within each byte shall not be changed. A BYTORD bit of one specifies all logical block(s) are transferred in the reverse order but the byte(s) within the logical block(s) are transferred in the same order as returned by the READ(16) command. Support for either value of the BYTORD bit is optional.

Refer to the READ(16) command (see 5.3) for a description of the FIXED bit, the SILI bit, the TRANSFER LENGTH field, and any conditions that may result from incorrect usage of these fields.

The LOGICAL OBJECT IDENTIFIER and PARTITION fields specify the position at which the READ REVERSE(16) command shall start. If the TRANSFER LENGTH field is not set to zero and the current logical position does not match the specified LOGICAL OBJECT IDENTIFIER and PARTITION fields, the device server shall perform a locate operation to the specified logical object identifier and partition prior to performing the read reverse operation. If the locate operation fails, the device server shall return CHECK CONDITION status and the additional sense code shall be set to LOCATE OPERATION FAILURE. The INFORMATION field in the sense data shall be set to the requested transfer length. Following a locate operation failure the logical position is undefined.

Filemarks, incorrect-length logical blocks, and unrecovered read errors are handled the same as in the READ(16) command, except that upon termination the logical position shall be before the filemark, incorrect-length logical block, or unrecovered logical block (beginning-of-partition side).

If the device server encounters beginning-of-partition during a READ REVERSE(16) command, CHECK CONDITION status shall be returned and the EOM and VALID bits shall be set to one in the sense data. The sense key shall be set to NO SENSE or RECOVERED ERROR, as appropriate. If the FIXED bit is one, the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks transferred. If the FIXED bit is zero, the INFORMATION field shall be set to the requested transfer length.

5.5 VERIFY(16) command

The VERIFY(16) command (see table 26) requests that the device server verify one or more logical block(s) beginning at the logical object identifier and partition specified in the command descriptor block. Prior to performing the verify operation, the device server shall perform a synchronize operation (see 4.2.10).

Table 26 — VERIFY(16) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (8Fh)							
1	Reserved					IMMED	BYTCMP	FIXED
2	Reserved							
3	PARTITION							
4	(MSB)							
5								
6								
7								
8				LOGICAL OBJECT IDENTIFIER				
9								
10								
11								(LSB)
12	(MSB)							
13				VERIFICATION LENGTH				
14								(LSB)
15	CONTROL							

An immediate (IMMED) bit of zero specifies the command shall not return status until the verify operation has completed. An IMMED bit of one specifies status shall be returned as soon as the command descriptor block has been validated; but after all verification data has been transferred from the application client to the device server, if the BYTCMP bit is one.

NOTE 22 In order to ensure that no errors are lost, the application client should set the IMMED bit to zero on the last VERIFY(16) command when issuing a series of VERIFY(16) commands.

A byte compare (BYTCMP) bit of zero specifies the verification shall be simply a medium verification (e.g., CRC, ECC). No data shall be transferred from the application client to the device server.

A BYTCMP bit of one specifies the device server shall perform a byte-by-byte compare of the data on the medium and the data transferred from the application client. Data shall be transferred from the application client to the device server as in a WRITE(16) command (see 5.6). If the BYTCMP bit is one and the byte compare option is not supported, the device server shall terminate the command with CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to INVALID FIELD IN CDB.

The LOGICAL OBJECT IDENTIFIER and PARTITION fields specify the position where the VERIFY(16) command shall start. If the VERFICATION LENGTH field is not zero and the current logical position does not match the specified LOGICAL OBJECT IDENTIFIER and PARTITION fields, the device server shall perform a locate operation to the specified logical object identifier and partition prior to performing the verify operation. If the locate operation fails, the device server shall return CHECK CONDITION status and the additional sense code shall be set to LOCATE OPERATION FAILURE. The INFORMATION field in the sense data shall be set to the requested verification length. Following a locate operation failure the logical position is undefined.

The VERIFICATION LENGTH field specifies the amount of data to verify, in logical blocks or bytes, as specified by the FIXED bit. Refer to the READ(16) command (see 5.3) for a description of the FIXED bit and any error conditions that may result from incorrect usage. If the VERIFICATION LENGTH field is zero, no data shall be verified and the current logical position shall not be changed. This condition shall not be considered an error.

The VERIFY(16) command shall terminate as follows:

a) when the verification length has been satisfied;
b) when an incorrect-length logical block is encountered;
c) when a filemark is encountered;
d) when end-of-data is encountered;
e) when the end-of-partition is encountered;
f) when early-warning is encountered (if the REW bit is one in the Device Configuration mode page, see 8.3.3); or
g) when an unrecoverable read error is encountered.

The status and sense data for each of the termination conditions are handled in the same manner as in the READ(16) command (see 5.3). Upon successful completion of a VERIFY(16) command, the logical position shall be after the last logical block verified (end-of-partition side).

If the data does not compare (BYTCMP bit of one), the command shall terminate with CHECK CONDITION status, the sense data VALID bit shall be set to one, the sense key shall be set to MISCOMPARE, and the additional sense code shall be set to MISCOMPARE DURING VERIFY OPERATION. If the FIXED bit is one, the INFORMATION field shall be set to the requested verification length minus the actual number of logical blocks successfully verified. If the FIXED bit is zero, the INFORMATION field shall be set to the requested verification length minus the actual number of bytes successfully verified. This number may be larger than the requested verification length if the error occurred on a previous VERIFY(16) command with an IMMED bit of one. Upon termination, the medium shall be positioned after the logical block containing the miscompare (end-of-partition side).

5.6 WRITE(16) command

The WRITE(16) command (see table 27) requests that the device server write the logical block that is transferred from the application client to the logical object identifier and partition specified in the command descriptor block.

Table 27 — WRITE(16) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (8Ah)							
1	Reserved				FCS	LCS	Rsvd	FIXED
2	Reserved							
3	PARTITION							
4	(MSB)							
5								
6								
7								
8				LOGICAL OBJECT IDENTIFIER				
9								
10								
11								(LSB)
12	(MSB)							
13				TRANSFER LENGTH				
14								(LSB)
15	CONTROL							

A first command in sequence (FCS) bit of one specifies this command is the first command in a tagged write sequence. An FCS bit of zero specifies this command is not the first command in a tagged write sequence.

A last command in sequence (LCS) bit of one specifies this command is the last command in a tagged write sequence. An LCS bit of zero specifies this command is not the last command in a tagged write sequence.

The FIXED bit specifies whether fixed-block transfers or variable-block tansfers are to be used. See the READ BLOCK LIMITS command (see 7.5) for additional information about fixed-block transfers and variable-block transfers.

If the FIXED bit is one, the TRANSFER LENGTH value specifies the number of fixed-length blocks to be transferred, using the current block length reported in the mode parameter block descriptor (see 8.3). If the FIXED bit is zero, a single logical block is transferred with TRANSFER LENGTH specifying the logical block length in bytes.

The LOGICAL OBJECT IDENTIFIER and PARTITION fields specify the position where the WRITE(16) command shall start. If the TRANSFER LENGTH field is not set to zero and the current logical position does not match the specified LOGICAL OBJECT IDENTIFIER and PARTITION fields, the device server shall perform a locate operation to the specified logical object identifier and partition prior to performing the write operation. If the locate operation fails, the device server shall return CHECK CONDITION status and the additional sense code shall be set to LOCATE OPERATION FAILURE. The INFORMATION field in the sense data shall be set to the requested transfer length. Following a locate operation failure the logical position is undefined.

If the TRANSFER LENGTH field is zero, no data shall be transferred and the current logical position shall not be changed. This condition shall not be considered an error.

A WRITE(16) command may be buffered or unbuffered, as specified by the BUFFERED MODE field of the mode parameter header (see 8.3). When operating in unbuffered mode (see 3.1.79), the device server shall not return GOOD status until all logical block(s) are successfully written to the medium. When operating in buffered mode (see 3.1.8), the device server may return GOOD status as soon as all logical block(s) are successfully transferred to the logical unit's object buffer.

NOTE 23 For compatibility with devices implemented prior to this version of this International Standard, a WRITE FILEMARKS(16) command with the IMMED bit set to zero should be issued when completing a buffered write operation to perform a synchronize operation (see 4.2.10).

If the device server enables a WRITE(16) command while positioned between EW and EOP, or encounters EW during the processing of a WRITE(16) command, an attempt to finish writing any data may be made as determined by the current settings of the REW and SEW bits in the Device Configuration mode page (see 8.3.3). The command shall terminate with CHECK CONDITION status and the additional sense code shall be set to END-OF-PARTITION/MEDIUM DETECTED. If all data that is to be written is successfully transferred to the medium, the sense key shall be set to NO SENSE or RECOVERED ERROR, as appropriate. If the device server is unable to transfer any data, buffered or unbuffered, when early-warning is encountered, the sense key shall be set to VOLUME OVERFLOW. If the SEW bit is set to zero, the EOM bit shall be set to one in the sense data. If the SEW bit is set to one, the EOM and VALID bits shall be set to one in the sense data.

The INFORMATION field shall be set as follows:

a) if the FIXED bit is set to one, the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks transferred to the device server; or
b) if the FIXED bit is set to zero, the INFORMATION field shall be set to the requested transfer length.

The device server should perform a synchronize operation (see 4.2.10) after the first early-warning indication has been returned to the application client (see 4.2.4).

NOTE 24 For some application clients it is important to recognize an error if end-of-partition is encountered during the processing of a WRITE(16) command, without regard for whether all data that is to be written is successfully transferred to the medium. The VOLUME OVERFLOW sense key may always validly be returned if end-of-partition is encountered while writing, and such usage is recommended. Reporting the MEDIUM ERROR sense key may cause confusion as to whether there was really defective medium encountered during the processing of the last WRITE(16) command.

If a WRITE(16) command is terminated early, an incomplete logical block (i.e., a logical block not completely transferred to the device server from the initiator) shall be discarded. The incomplete logical block may be accessible prior to new data being written to the media. The device server shall be logically positioned after the last logical block that was successfully transferred.

See 4.2.5 for a description of the effect of a PEWZ on the completion of the WRITE(16) command.

5.7 WRITE FILEMARKS(16) command

The WRITE FILEMARKS(16) command (see table 28) requests that the device server write the specified number of filemarks to the logical object identifier and partition specified in the command descriptor block.

Table 28 — WRITE FILEMARKS(16) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (80h)							
1	Reserved				FCS	LCS	Obsolete	IMMED
2	Reserved							
3	PARTITION							
4	(MSB)							
5								
6								
7								
8				LOGICAL OBJECT IDENTIFIER				
9								
10								
11								(LSB)
12	(MSB)							
13				FILEMARK COUNT				
14								(LSB)
15	CONTROL							

A first command in sequence (FCS) bit of one specifies this command is the first command in a tagged write sequence. An FCS bit of zero specifies this command is not the first command in a tagged write sequence.

A last command in sequence (LCS) bit of one specifies this command is the last command in a tagged write sequence. An LCS bit of zero specifies this command is not the last command in a tagged write sequence.

An immediate (IMMED) bit of one specifies the device server shall return status as soon as the command descriptor block has been validated. An IMMED bit of one is only valid if the device is operating in buffered mode (see 3.1.8). If the IMMED bit is set to one and the device is operating in unbuffered mode the command shall be terminated with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB.

An IMMED bit of zero specifies the device server shall not return status until the write operation has completed. Any buffered logical objects shall be written to the medium prior to completing the command.

NOTE 25 Upon completion of any buffered write operation, the application client may issue a WRITE FILEMARKS(16) command with the IMMED bit set to zero and the FILEMARK COUNT field set to zero to perform a synchronize operation (see 4.2.10).

The LOGICAL OBJECT IDENTIFIER and PARTITION fields specify the position where the WRITE FILEMARKS(16) command shall start. If the FILEMARK COUNT field is not set to zero and the current logical position does not match

the specified LOGICAL OBJECT IDENTIFIER and PARTITION fields, the device server shall perform a locate operation to the specified logical object identifier and partition prior to performing the write filemarks operation. If the locate operation fails, the device server shall return CHECK CONDITION status and the additional sense code shall be set to LOCATE OPERATION FAILURE. The INFORMATION field in the sense data shall be set to the requested filemark count. Following a locate operation failure the logical position is undefined.

The FILEMARK COUNT field specifies the number of filemarks to be written. If the FILEMARK COUNT field is set to zero, the current logical position shall not be changed. It shall not be considered an error if the FILEMARK COUNT field is set to zero.

NOTE 26 The FILEMARK COUNT field takes the place of the TRANSFER LENGTH field normally used by medium access commands that move data from the Data-Out Buffer to the device server (see SPC-4).

If the device server enables a WRITE FILEMARKS(16) command while positioned between EW and EOP, or encounters EW during the processing of a WRITE FILEMARKS(16) command, an attempt to finish writing any buffered logical objects may be made, as determined by the current settings of the REW and SEW bits in the Device Configuration mode page (see 8.3.3). The command shall terminate with CHECK CONDITION status and the additional sense code shall be set to END-OF-PARTITION/MEDIUM DETECTED. If all buffered logical objects to be written are successfully transferred to the medium, the sense key shall be set to NO SENSE or RECOVERED ERROR, as appropriate. If the device server is unable to transfer any buffered logical object when early-warning is encountered, the sense key shall be set to VOLUME OVERFLOW. If the SEW bit is set to zero, the EOM bit shall be set to one in the sense data. If the SEW bit is set to one, the EOM and VALID bits shall be set to one in the sense data.

The INFORMATION field shall be set to the FILEMARK COUNT field value minus the actual number of filemarks:

a) written to the object buffer if the IMMED bit is set to one; or
b) written to the medium if the IMMED bit is set to zero.

The device server should perform a synchronize operation (see 4.2.10) after the first early-warning indication has been returned to the application client (see 4.2.4).

See 4.2.5 for a description of the effect of a PEWZ on the completion of the WRITE FILEMARKS(16) command.

6 Implicit address command descriptions for sequential-access devices

6.1 Summary of commands for implicit address mode

The implicit address commands for sequential-access devices are shown in table 29. Commands specified as mandatory in table 29 shall be implemented if the implicit address command set is supported.

If a synchronize operation is required for a command, the synchronize operation shall be performed as specified in 4.2.10.

Refer to table 14 for a description of reservations.

The following command codes are vendor specific: 02h, 06h, 07h, 09h, 0Ch, 0Dh, and 0Eh.

Table 29 — Implicit address command set for sequential-access devices

Command Name	Type	OpCode	Synchronize Operation Required[a]	Reference
ACCESS CONTROL IN	O	86h	No	SPC-4
ACCESS CONTROL OUT	O	87h	No	SPC-4
CHANGE ALIAS	O	A4h/0Bh[b]	No	SPC-4
ERASE(6)	M	19h	Yes	6.2
EXTENDED COPY	O	83h	No	SPC-4
FORMAT MEDIUM	O	04h	No	7.1
INQUIRY	M	12h	No	SPC-4
LOAD UNLOAD	O	1Bh	Yes	7.2
LOCATE(10)	M	2Bh	Yes	6.3
LOCATE(16)	M	92h	Yes	7.3
LOG SELECT	O	4Ch	No	SPC-4
LOG SENSE	O	4Dh	No	SPC-4
MODE SELECT(6)	M	15h	Yes[a]	SPC-4
MODE SELECT(10)	O	55h	Yes[a]	SPC-4
MODE SENSE(6)	M	1Ah	No	SPC-4
MODE SENSE(10)	O	5Ah	No	SPC-4
PERSISTENT RESERVE IN	M	5Eh	No	SPC-4

Key: M = Command implementation is mandatory.
O = Command implementation is optional.
X = See referenced standard or subclause.
Z = Command implementation is specified in a previous standard.

a. Refer to 4.2.10.
b. This command is defined by a combination of operation code and service action. The operation code value is shown preceding the slash and the service action value is shown after the slash.

Table 29 — Implicit address command set for sequential-access devices (Continued)

Command Name	Type	OpCode	Synchronize Operation Required[a]	Reference
PERSISTENT RESERVE OUT	M	5Fh	No	SPC-4
PREVENT ALLOW MEDIUM REMOVAL	O	1Eh	No	SPC-4
READ(6)	M	08h	Yes	6.4
READ ATTRIBUTE	O	8Ch	No	SPC-4
READ BLOCK LIMITS	M	05h	No	7.5
READ BUFFER	O	3Ch	Yes	SPC-4
READ POSITION	M	34h	No	7.6
READ REVERSE(6)	O	0Fh	Yes	6.5
RECEIVE COPY RESULTS	O	84h	No	SPC-4
RECEIVE DIAGNOSTIC RESULTS	O	1Ch	No	SPC-4
RECOVER BUFFERED DATA	O	14h	May	7.7
REPORT DENSITY SUPPORT	M	44h	No	7.8
REPORT ALIAS	O	A3h/0Bh[b]	No	SPC-4
REPORT DEVICE IDENTIFIER	O	A3h/05h[b]	No	SPC-4
REPORT LUNS	X	A0h	No	SPC-4
REPORT SUPPORTED OPERATION CODES	O	A3h/0Ch[b]	No	SPC-4
REPORT TARGET PORT GROUPS	O	A3h/0Ah[b]	No	SPC-4
REQUEST SENSE	M	03h	No	SPC-4
REWIND	M	01h	Yes	7.9
SECURITY PROTOCOL IN	O	A2h	No	SPC-4
SECURITY PROTOCOL OUT	O	B5h	No	SPC-4
SEND DIAGNOSTIC	M	1Dh	Yes[a]	SPC-4
SET CAPACITY	O	0Bh	May	7.10
SET DEVICE IDENTIFIER	O	A4h/06h[b]	No	SPC-4
SET TARGET PORT GROUPS	O	A4h/0Ah[b]	No	SPC-4
SPACE(6)	M	11h	May	6.6
SPACE(16)	O	91h	May	7.11

Key: M = Command implementation is mandatory.
O = Command implementation is optional.
X = See referenced standard or subclause.
Z = Command implementation is specified in a previous standard.

a. Refer to 4.2.10.
b. This command is defined by a combination of operation code and service action. The operation code value is shown preceding the slash and the service action value is shown after the slash.

Table 29 — Implicit address command set for sequential-access devices (Continued)

Command Name	Type	OpCode	Synchronize Operation Required[a]	Reference
TEST UNIT READY	M	00h	No	SPC-4
VERIFY(6)	O	13h	Yes	6.7
WRITE(6)	M	0Ah	No	6.8
WRITE ATTRIBUTE	O	8Dh	No	SPC-4
WRITE BUFFER	O	3Bh	Yes[a]	SPC-4
WRITE FILEMARKS(6)	M	10h	May	6.9
Obsolete	Z	A7h		
Obsolete	Z	B4h		

Key: M = Command implementation is mandatory.
O = Command implementation is optional.
X = See referenced standard or subclause.
Z = Command implementation is specified in a previous standard.

a. Refer to 4.2.10.
b. This command is defined by a combination of operation code and service action. The operation code value is shown preceding the slash and the service action value is shown after the slash.

6.2 ERASE(6) command

The ERASE(6) command (see table 30) causes part or all of the medium to be erased beginning at the current position. Prior to performing the erase operation, the device server shall perform a synchronize operation (see 4.2.10).

Table 30 — ERASE(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (19h)							
1	Reserved						IMMED	LONG
2	Reserved		METHOD		Reserved		SMD	VCM
3	Reserved							
4	Reserved							
5	CONTROL							

An immediate (IMMED) bit of zero specifies the device server shall not return status until the erase operation has completed. Interpretation of an IMMED bit of one depends on the value of the LONG bit, see below. However, for all values of the LONG bit, if CHECK CONDITION status is returned for an ERASE(6) command with an IMMED bit of one, the erase operation shall not be performed.

NOTE 27 Application clients should set the IMMED bit set to zero to guarantee the operation has completed successfully when setting the METHOD field to 10b. When the METHOD field is set to 10b, the duration of the processing may be extended (i.e., may be longer than just setting the LONG bit to one).

A LONG bit of one specifies all remaining medium in the current partition beginning at the current logical position shall be erased using the method specified by the METHOD field value (see table 23). If the format on the medium specifies a recorded indication of EOD (see 3.1.18), the erase operation shall establish an EOD indication at the current logical position as part of the erase operation. If the IMMED bit is one, the device server shall return status as soon as all buffered logical objects have been written to the medium and the command descriptor block of the ERASE(6) command has been validated. The logical position following an ERASE(6) command with a LONG bit of one is not specified by this standard.

NOTE 28 Some logical units may reject an ERASE(6) command if the logical unit is not at beginning-of-partition.

A LONG bit of zero specifies the device server shall perform the action specified by the SHORT ERASE MODE field in the Device Configuration Extension mode page (see 8.3.8) at the current logical position. If the IMMED bit is one, the device server shall return status as soon as the command descriptor block has been validated.

If the logical unit encounters early-warning during an ERASE(6) command, and any buffered logical objects remain to be written, the device server action shall be as defined for the early-warning condition of the WRITE(6) command (see 6.8). If the LONG bit is zero, the erase operation shall terminate with CHECK CONDITION status and set the sense data as defined for the WRITE(6) command. Any count of pending buffered erases shall not be reported as part of the value returned in the INFORMATION field or in the READ POSITION response data.

The METHOD field, Security Meta-Data (SMD) bit, and the Vendor-specific Control Meta-data (VCM) bit are defined in 5.2.

See 4.2.5 for a description of the effect of a PEWZ on the completion of the ERASE(6) command.

6.3 LOCATE(10) command

The LOCATE(10) command (see table 31) causes the logical unit to position the medium to the specified logical object with a matching logical object identifier in the specified partition. Upon completion, the logical position shall be before the specified logical object. Prior to performing the locate operation, the device server shall perform a synchronize operation (see 4.2.10).

Table 31 — LOCATE(10) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (2Bh)							
1	Reserved					BT	CP	IMMED
2	Reserved							
3	(MSB)							
4				LOGICAL OBJECT IDENTIFIER				
5								
6								(LSB)
7	Reserved							
8	PARTITION							
9	CONTROL							

A block identifier type (BT) bit of one specifies the value in the LOGICAL OBJECT IDENTIFIER field shall be interpreted as a vendor-specific value. A BT bit of zero specifies the value in the LOGICAL OBJECT IDENTIFIER field shall be interpreted as a logical object identifier (see 3.1.41).

A change partition (CP) bit of one specifies a change to the partition specified in the PARTITION field shall occur prior to positioning to the logical object specified in the LOGICAL OBJECT IDENTIFIER field. A CP bit of zero specifies no partition change shall occur and the PARTITION field shall be ignored.

An immediate (IMMED) bit of zero specifies the device server shall not return status until the locate operation has completed. If the IMMED bit is one, the device server shall return status as soon as all buffered logical objects have been written to the medium and the command descriptor block of the LOCATE(10) command has been validated. If CHECK CONDITION status is returned for a LOCATE(10) command with an IMMED bit of one, the locate operation shall not be performed.

The LOGICAL OBJECT IDENTIFIER field specifies the logical object identifier to which the logical unit shall position the medium based on the current setting of the BT bit. An otherwise valid LOCATE(10) command to any position between beginning-of-data and the position immediately after the last block in the partition (position at end-of-data) shall not return a sense key of ILLEGAL REQUEST. A LOCATE(10) to a position past end-of-data shall return CHECK CONDITION status and the sense key shall be set to BLANK CHECK. Additionally, the sense data EOM bit shall be set to one if end-of-data is located at or after early-warning.

If end-of-partition is encountered, CHECK CONDITION status shall be returned, the sense key shall be set to MEDIUM ERROR, and the sense data EOM bit shall be set to one.

The PARTITION field specifies the partition to select if the CP bit is one. Refer to the sequential-access device model (see 4.2.6) and the Medium Partition mode page (see 8.3.4) for additional information about partitioning.

The logical unit position is undefined if a LOCATE(10) command fails with a sense key other than ILLEGAL REQUEST.

6.4 READ(6) command

The READ(6) command (see table 32) requests that the device server transfer one or more logical block(s) to the application client beginning with the next logical block. Prior to performing the read operation, the device server shall perform a synchronize operation (see 4.2.10).

Table 32 — READ(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (08h)							
1	Reserved						SILI	FIXED
2	(MSB)							
3	TRANSFER LENGTH							
4								(LSB)
5	CONTROL							

The FIXED bit specifies whether fixed-block transfers or variable-block transfers are to be used. Refer to the READ BLOCK LIMITS command (see 7.5) for additional information about fixed-block transfers and variable-block transfers.

If the FIXED bit is one, the TRANSFER LENGTH specifies the number of fixed-length blocks to be transferred, using the current block length reported in the mode parameters block descriptor (see 8.3). If the FIXED bit is zero, a variable-length block is requested with the TRANSFER LENGTH specifying the maximum number of bytes allocated for the returned logical block.

A successful READ(6) command with a FIXED bit of one shall transfer the requested transfer length times the current block length in bytes to the application client. A successful READ(6) command with a FIXED bit of zero shall transfer the requested transfer length in bytes to the application client. Upon completion, the logical position shall be after the last logical block transferred (end-of-partition side).

If the suppress incorrect-length indicator (SILI) bit is one and the FIXED bit is zero, the device server shall:

a) report CHECK CONDITION status for an incorrect-length condition only if the overlength condition exists and the BLOCK LENGTH field in the mode parameter block descriptor is nonzero (see 8.3); or
b) not report CHECK CONDITION status if the only error is the underlength condition, or if the only error is the overlength condition and the BLOCK LENGTH field of the mode parameters block descriptor is zero.

NOTE 29 Since the residual information normally provided in the INFORMATION field of the sense data may not be available when the SILI bit is set, other methods for determining the actual block length should be used (e.g., including length information in the logical block).

If the SILI bit is one and the FIXED bit is one, the device server shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB.

If the SILI bit is zero and an incorrect-length logical block is read, CHECK CONDITION status shall be returned. The ILI and VALID bits shall be set to one in the sense data and the additional sense code shall be set to NO ADDITIONAL SENSE INFORMATION. Upon termination, the logical position shall be after the incorrect-length logical block (end-of-partition side). If the FIXED bit is one, the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks read (not including the incorrect-length logical block). If the FIXED bit is zero, the INFORMATION field shall be set to the requested transfer length minus the actual logical block length. Logical units that do not support negative values shall set the INFORMATION field to zero if the overlength condition exists.

NOTE 30 In the above case with the FIXED bit of one, only the position of the incorrect-length logical block may be determined from the sense data. The actual length of the incorrect logical block is not reported. Other means may be used to determine its actual length (e.g., read it again with the fixed bit set to zero).

A TRANSFER LENGTH of zero specifies no data shall be transferred, and the logical position shall not be changed. This condition shall not be considered an error.

In the case of an unrecovered read error, if the FIXED bit is one, the sense data VALID bit shall be set to one and the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks read (not including the unrecovered logical blocks). If the FIXED bit is zero, the sense data VALID bit shall be set to one and the INFORMATION field shall be set to the requested transfer length. Upon termination, the logical position shall be after the unrecovered logical block.

If the device server encounters a filemark during a READ(6) command, CHECK CONDITION status shall be returned and the FILEMARK and VALID bits shall be set to one in the sense data. The sense key shall be set to NO SENSE or RECOVERED ERROR, as appropriate, and the additional sense code shall be set to FILEMARK DETECTED. Upon termination, the logical position shall be after the filemark (end-of-partition side). If the FIXED bit is one, the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks read. If the FIXED bit is zero, the INFORMATION field shall be set to the requested transfer length.

If the device server encounters early-warning during a READ(6) command and the REW bit is set to one in the Device Configuration mode page (see 8.3.3), CHECK CONDITION status shall be returned upon completion of the current logical block. The sense key shall be set to NO SENSE or RECOVERED ERROR, as appropriate, and the additional sense code shall be set to END-OF-PARTITON/MEDIUM DETECTED. The EOM and VALID bits shall be set to one in the sense data. Upon termination, the logical position shall be after the last logical block transferred (end-of-partition side). If the FIXED bit is one, the INFORMATION field in the sense data shall be set to the requested transfer length minus the actual number of logical blocks read. If the FIXED bit is zero, the INFORMATION field in the sense data shall be set to the requested transfer length minus the actual logical block length. The device server shall not return CHECK CONDITION status when early-warning is encountered if the REW bit is zero.

NOTE 31 A REW bit of one is not recommended for most applications since read data may be present after early-warning.

If the device server encounters end-of-data during a READ(6) command, CHECK CONDITION status shall be returned, the sense key shall be set to BLANK CHECK, and the VALID bit shall be set to one in the sense data. If end-of-data is encountered at or after early-warning, the EOM bit shall also be set to one. Upon termination, the logical position shall be immediately after the last recorded logical object (end-of-partition side). If the FIXED bit is one, the INFORMATION field in the sense data shall be set to the requested transfer length minus the actual number of logical blocks read. If the FIXED bit is zero, the INFORMATION field shall be set to the requested transfer length.

If the device server encounters end-of-partition during a READ(6) command, CHECK CONDITION status shall be returned, the sense key shall be set to MEDIUM ERROR, and the EOM and VALID bits shall be set to one in the sense data. The medium position following this condition is not defined. If the FIXED bit is one, the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks read. If the FIXED bit is zero, the INFORMATION field in the sense data shall be set to the requested transfer length.

6.5 READ REVERSE(6) command

The READ REVERSE(6) command (see table 33) requests that the device server transfer one or more logical block(s) to the application client beginning at the current logical position. Prior to performing the read reverse operation, the device server shall perform a synchronize operation (see 4.2.10).

Table 33 — READ REVERSE(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (0Fh)							
1	Reserved					BYTORD	SILI	FIXED
2	(MSB)							
3				TRANSFER LENGTH				
4								(LSB)
5	CONTROL							

This command is similar to the READ(6) command except that medium motion is in the reverse direction. Upon completion of a READ REVERSE(6) command, the logical position shall be before the last logical block transferred (beginning-of-partition side).

A byte order (BYTORD) bit of zero specifies all logical block(s), and the byte(s) within the logical block(s), are transferred in the reverse order. The order of bits within each byte shall not be changed. A BYTORD bit of one specifies all logical block(s) are transferred in the reverse order but the byte(s) within the logical block(s) are transferred in the same order as returned by the READ(6) command. Support for either value of the BYTORD bit is optional.

Refer to the READ(6) command (see 6.4) for a description of the FIXED bit, the SILI bit, the TRANSFER LENGTH field, and any conditions that may result from incorrect usage of these fields.

Filemarks, incorrect-length logical blocks, and unrecovered read errors are handled the same as in the READ(6) command, except that upon termination the logical position shall be before the filemark, incorrect-length logical block, or unrecovered block (beginning-of-partition side).

If the device server encounters beginning-of-partition during a READ REVERSE(6) command, CHECK CONDITION status shall be returned and the EOM and VALID bits shall be set to one in the sense data. The sense key shall be set to NO SENSE or RECOVERED ERROR, as appropriate. If the FIXED bit is one, the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks transferred. If the FIXED bit is zero, the INFORMATION field shall be set to the requested transfer length.

6.6 SPACE(6) command

The SPACE(6) command (see table 34) provides a variety of positioning functions that are determined by the CODE and COUNT fields. Both forward and reverse positioning are provided, although some logical units may only support a subset of this command. Prior to performing the space operation, except as stated in the description of the COUNT field, the device server shall perform a synchronize operation (see 4.2.10). If an application client requests an unsupported function, the command shall be terminated with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB. The INFORMATION field value shall be equal to the magnitude of the COUNT field minus the magnitude of the logical objects

spaced over. A CHECK CONDITION caused by early termination of any SPACE(6) command shall not result in a negative INFORMATION field value.

Table 34 — SPACE(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (11h)							
1	Reserved				CODE			
2	(MSB)							
3				COUNT				
4								(LSB)
5	CONTROL							

The CODE field is defined in table 35.

Table 35 — Code definition

Code	Description	Support
0000b	Logical blocks	M
0001b	Filemarks	M
0010b	Sequential filemarks	O
0011b	End-of-data	O
0100b	Obsolete	
0101b	Obsolete	
0110b-1111b	Reserved	

When spacing over logical objects, the COUNT field specifies the number of logical objects to be spaced over in the current partition. A positive value *N* in the COUNT field when the CODE field is not 0011b (End-of-data) shall cause forward positioning (toward end-of-partition) over *N* logical objects ending on the end-of-partition side of the last logical object, if they exist. A zero value in the COUNT field when the CODE field is not 0011b (End-of-data) shall cause no change of logical position. A negative value -*N* (two's complement notation) in the COUNT field when the CODE field is not 0011b (End-of-data) shall cause reverse positioning (toward beginning-of-partition) over *N* logical objects ending on the beginning-of-partition side of the last logical object, if they exist. When the CODE field is 0011b (End-of-data), the COUNT field shall be ignored and the device server shall perform a synchronize operation before moving before the end-of-data position. When the COUNT field is zero and the CODE field is not 0011b (End-of-data), a device server is not required to perform a synchronize operation. Support of spacing in the reverse direction is optional.

If a filemark is encountered while spacing over logical blocks, the command shall be terminated. CHECK CONDITION status shall be returned, and the FILEMARK and VALID bits shall be set to one in the sense data. The sense key shall be set to NO SENSE and the additional sense code shall be set to FILEMARK DETECTED. The INFORMATION field shall be set to the requested count minus the actual number of logical blocks spaced over. The logical position shall be on the end-of-partition side of the filemark if movement was in the forward direction and on the beginning-of-partition side of the filemark if movement was in the reverse direction.

If early-warning is encountered while spacing over logical objects and the REW bit is set to one in the Device Configuration mode page (see 8.3.3), CHECK CONDITION status shall be returned, the sense key shall be set to

NO SENSE, and the EOM and VALID bits shall be set to one in the sense data. The additional sense code shall be set to END-OF-PARTITION/MEDIUM DETECTED. The INFORMATION field shall be set to the requested count minus the actual number of logical objects spaced over as defined by the CODE value. If the REW bit is zero, the device server shall not report CHECK CONDITION status at the early-warning point.

NOTE 32 Setting the REW bit to one is not recommended for most applications since data may be present after early-warning.

If end-of-data is encountered while spacing over logical objects, CHECK CONDITION status shall be returned, the sense key shall be set to BLANK CHECK, and the sense data VALID bit shall be set to one in the sense data. The additional sense code shall be set to END-OF-DATA DETECTED. The sense data EOM bit shall be set to one if end-of-data is encountered at or after early-warning. The INFORMATION field shall be set to the requested count minus the actual number of logical objects spaced over as defined by the CODE value. The medium shall be positioned such that a subsequent write operation would append to the last logical object.

If the end-of-partition is encountered while spacing forward over logical objects, CHECK CONDITION status shall be returned, and the sense key shall be set to MEDIUM ERROR. The additional sense code shall be set to END-OF-PARTITION/MEDIUM DETECTED, and the sense data EOM and VALID bit shall be set to one. The INFORMATION field shall be set to the requested count minus the actual number of logical objects spaced over as defined by the CODE value. The medium position following this condition is not defined.

If beginning-of-partition is encountered while spacing over logical objects in the reverse direction, the device server shall return CHECK CONDITION status and shall set the sense key to NO SENSE. The additional sense code shall be set to BEGINNING-OF-PARTITION/MEDIUM DETECTED. The sense data EOM and VALID bits shall be set to one, and the INFORMATION field set to the total number of logical objects not spaced over as defined by the CODE value (i.e., the requested number of logical objects minus the actual number of logical objects spaced over as defined by the CODE value). The medium position following this condition is not defined. A successfully completed SPACE(6) command shall not set EOM to one at beginning-of-partition.

When spacing over sequential filemarks, the count field is interpreted as follows:

a) a positive value *N* shall cause forward movement to the first occurrence of *N* or more consecutive filemarks being logically positioned after the N^{th} filemark;
b) a zero value shall cause no change in the logical position; or
c) a negative value *-N* (2's complement notation) shall cause reverse movement to the first occurrence of *N* or more consecutive filemarks being logically positioned on the beginning-of-partition side of the N^{th} filemark.

If end-of-partition is encountered while spacing to sequential filemarks, CHECK CONDITION status shall be returned, and the sense key shall be set to MEDIUM ERROR. The additional sense code shall be set to END-OF-PARTITION/MEDIUM DETECTED, the EOM bit shall be set to one, and the VALID bit shall be set to zero in the sense data. The medium position following this condition is not defined.

If end-of-data is encountered while spacing to sequential filemarks, CHECK CONDITION status shall be returned, and the sense key shall be set to BLANK CHECK. The additional sense code shall be set to END-OF-DATA DETECTED, and the sense data VALID bit shall be set to zero. The medium shall be positioned such that a subsequent write operation would append to the last logical object. The sense data EOM bit shall be set to one if end-of-data is encountered at or after early-warning.

When spacing to end-of-data, the COUNT field is ignored. Upon successful completion, the medium shall be positioned such that a subsequent write operation would append to the last logical object.

If end-of-partition is encountered while spacing to end-of-data, CHECK CONDITION status shall be returned, and the sense key shall be set to MEDIUM ERROR. The additional sense code shall be set to END-OF-PARTITION/

MEDIUM DETECTED, the EOM bit shall be set to one, and the VALID bit shall be set to zero in the sense data. The medium position following this condition is not defined.

6.7 VERIFY(6) command

The VERIFY(6) command (see table 36) requests that the device server verify one or more logical block(s) beginning at the current logical position. Prior to performing the verify operation, the device server shall perform a synchronize operation (see 4.2.10).

Table 36 — VERIFY(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (13h)							
1	Reserved					IMMED	BYTCMP	FIXED
2	(MSB)							
3				VERIFICATION LENGTH				
4								(LSB)
5	CONTROL							

An immediate (IMMED) bit of zero specifies the command shall not return status until the verify operation has completed. An IMMED bit of one specifies status shall be returned as soon as the command descriptor block has been validated; but after all verification data has been transferred from the application client to the device server, if the BYTCMP bit is one.

NOTE 33 In order to ensure that no errors are lost, the application client should set the IMMED bit to zero on the last VERIFY(6) command when issuing a series of VERIFY(6) commands.

A byte compare (BYTCMP) bit of zero specifies the verification shall be simply a medium verification (e.g., CRC, ECC). No data shall be transferred from the application client to the device server.

A BYTCMP bit of one specifies the device server shall perform a byte-by-byte compare of the data on the medium and the data transferred from the application client. Data shall be transferred from the application client to the device server as in a WRITE(6) command (see 6.8). If the BYTCMP bit is one and the byte compare option is not supported, the device server shall terminate the command with CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to INVALID FIELD IN CDB.

The VERIFICATION LENGTH field specifies the amount of data to verify, in logical blocks or bytes, as specified by the FIXED bit. Refer to the READ(6) command (see 6.4) for a description of the FIXED bit and any error conditions that may result from incorrect usage. If the VERIFICATION LENGTH field is zero, no data shall be verified and the current logical position shall not be changed. This condition shall not be considered as an error.

The VERIFY(6) command shall terminate as follows:

a) when the verification length has been satisfied;
b) when an incorrect-length logical block is encountered;
c) when a filemark is encountered;
d) when end-of-data is encountered;
e) when the end-of-partition is encountered;

f) when early-warning is encountered (if the REW bit is one in the Device Configuration mode page, see 8.3.3); or
g) when an unrecoverable read error is encountered.

The status and sense data for each of the termination conditions are handled in the same manner as in the READ(6) command (see 6.4). Upon successful completion of a VERIFY(6) command, the logical position shall be after the last logical block verified.

If the data does not compare (BYTCMP bit of one), the command shall terminate with CHECK CONDITION status, the sense data VALID bit shall be set to one, the sense key shall be set to MISCOMPARE, and the additional sense code shall be set to MISCOMPARE DURING VERIFY OPERATION. If the FIXED bit is one, the INFORMATION field shall be set to the requested verification length minus the actual number of logical blocks successfully verified. If the FIXED bit is zero, the INFORMATION field shall be set to the requested verification length minus the actual number of bytes successfully verified. This number may be larger than the requested verification length if the error occurred on a previous VERIFY(6) command with an IMMED bit of one. Upon termination, the medium shall be positioned after the logical block containing the miscompare (end-of-partition side).

6.8 WRITE(6) command

The WRITE command (see table 37) requests that the device server write the logical block that is transferred from the application client to the current logical position.

Table 37 — WRITE(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (0Ah)							
1	Reserved							FIXED
2	(MSB)							
3				TRANSFER LENGTH				
4								(LSB)
5	CONTROL							

The FIXED bit specifies whether fixed-block transfers or variable-block transfers are to be used. See the READ BLOCK LIMITS command (see 7.5) for additional information about fixed-block transfers and variable-block transfers.

If the FIXED bit is one, the TRANSFER LENGTH value specifies the number of fixed-length blocks to be transferred, using the current block length reported in the mode parameter block descriptor (see 8.3). If the FIXED bit is zero, a single logical block is transferred with TRANSFER LENGTH specifying the logical block length in bytes.

If TRANSFER LENGTH is zero, no data shall be transferred and the current position shall not be changed. This condition shall not be considered an error.

A WRITE(6) command may be buffered or unbuffered, as specified by the BUFFERED MODE field of the mode parameter header (see 8.3). When operating in unbuffered mode (see 3.1.79), the device server shall not return GOOD status until all logical block(s) are successfully written to the medium. When operating in buffered mode (see 3.1.8), the device server may return GOOD status as soon as all logical block(s) are successfully transferred to the logical unit's object buffer.

NOTE 34 For compatibility with devices implemented prior to this version of this International Standard, a WRITE FILEMARKS command with the IMMED bit set to zero should be issued when completing a buffered write operation to perform a synchronize operation (see 4.2.10).

If the device server enables a WRITE(6) command while positioned between EW and EOP, or encounters EW during the processing of a WRITE(6) command, an attempt to finish writing any data may be made as determined by the current settings of the REW and SEW bits in the Device Configuration mode page (see 8.3.3). The command shall terminate with CHECK CONDITION status and the additional sense code shall be set to END-OF-PARTITION/MEDIUM DETECTED. If all data that is to be written is successfully transferred to the medium, the sense key shall be set to NO SENSE or RECOVERED ERROR, as appropriate. If the device server is unable to transfer any data, buffered or unbuffered, when early-warning is encountered, the sense key shall be set to VOLUME OVERFLOW. If the SEW bit is set to zero, the EOM bit shall be set to one in the sense data. If the SEW bit is set to one, the EOM and VALID bits shall be set to one in the sense data.

The INFORMATION field shall be set as follows:

a) if the FIXED bit is set to one, the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks transferred to the device server; or
b) if the FIXED bit is set to zero, the INFORMATION field shall be set to the requested transfer length.

The device server should perform a synchronize operation (see 4.2.10) after the first early-warning indication has been returned to the application client (see 4.2.4).

NOTE 35 For some application clients it is important to recognize an error if end-of-partition is encountered during the processing of a WRITE(6) command, without regard for whether all data that is to be written is successfully transferred to the medium. The VOLUME OVERFLOW sense key may always validly be returned if end-of-partition is encountered while writing, and such usage is recommended. Reporting the MEDIUM ERROR sense key may cause confusion as to whether there was really defective medium encountered during the processing of the last WRITE(6) command.

If a WRITE(6) command is terminated early, an incomplete logical block (i.e., a logical block not completely transferred to the device server from the initiator) shall be discarded. The incomplete logical block may be accessible prior to new data being written to the media. The device server shall be logically positioned after the last logical block that was successfully transferred.

See 4.2.5 for a description of the effect of a PEWZ on the completion of the WRITE(6) command.

6.9 WRITE FILEMARKS(6) command

The WRITE FILEMARKS(6) command (see table 38) requests that the device server write the specified number of filemarks to the current position.

Table 38 — WRITE FILEMARKS(6) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (10h)							
1	Reserved						Obsolete	IMMED
2	(MSB)							
3				FILEMARK COUNT				
4								(LSB)
5	CONTROL							

An immediate (IMMED) bit of one specifies the device server shall return status as soon as the command descriptor block has been validated. An IMMED bit of one is only valid if the device is operating in buffered mode (see 3.1.8). If the IMMED bit is set to one and the device is operating in unbuffered mode the command shall be terminated with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB.

An IMMED bit of zero specifies the device server shall not return status until the write operation has completed. Any buffered logical objects shall be written to the medium prior to completing the command.

The FILEMARK COUNT field specifies the number of filemarks to be written. If the FILEMARK COUNT field is set to zero, the current logical position shall not be changed. It shall not be considered an error if the FILEMARK COUNT field is set to zero.

NOTE 36 The FILEMARK COUNT field takes the place of the TRANSFER LENGTH field normally used by medium access commands that move data from the Data-Out Buffer to the device server (see SPC-4).

NOTE 37 Upon completion of any buffered write operation, the application client may issue a WRITE FILEMARKS(6) command with the IMMED bit set to zero and the FILEMARK COUNT field set to zero to perform a synchronize operation (see 4.2.10).

If the device server enables a WRITE FILEMARKS(6) command while positioned between EW and EOP, or encounters EW during the processing of a WRITE FILEMARKS(6) command, an attempt to finish writing any buffered logical objects may be made, as determined by the current settings of the REW and SEW bits in the Device Configuration mode page (see 8.3.3). The command shall terminate with CHECK CONDITION status and the additional sense code shall be set to END-OF-PARTITION/MEDIUM DETECTED. If all buffered logical objects to be written are successfully transferred to the medium, the sense key shall be set to NO SENSE or RECOVERED ERROR, as appropriate. If the device server is unable to transfer any buffered logical objects when early-warning is encountered, the sense key shall be set to VOLUME OVERFLOW. If the SEW bit is set to zero, the EOM bit shall be set to one in the sense data. If the SEW bit is set to one, the EOM and VALID bits shall be set to one in the sense data.

The INFORMATION field shall be set to the FILEMARK COUNT field value minus the actual number of filemarks:

a) written to the object buffer if the IMMED bit is set to one; or
b) written to the medium if the IMMED bit is set to zero.

The device server should perform a synchronize operation (see 4.2.10) after the first early-warning indication has been returned to the application client (see 4.2.4).

See 4.2.5 for a description of the effect of a PEWZ on the completion of the WRITE FILEMARKS(6) command.

7 Common command descriptions for sequential-access devices

7.1 FORMAT MEDIUM command

The FORMAT MEDIUM command (see table 39) is used to prepare the medium for use by the logical unit. If buffered logical objects are stored by the device server when processing of a FORMAT MEDIUM command begins, the command shall be rejected with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to POSITION PAST BEGINNING OF MEDIUM.

Table 39 — FORMAT MEDIUM command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (04h)							
1	Reserved						VERIFY	IMMED
2	Reserved						FORMAT	
3	(MSB)							
4	TRANSFER LENGTH							(LSB)
5	CONTROL							

The FORMAT MEDIUM command shall be accepted only when the medium is at beginning-of-partition 0 (BOP 0). If the medium is logically at any other position, the command shall be rejected with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to POSITION PAST BEGINNING OF MEDIUM.

At the successful completion of a FORMAT MEDIUM command, the medium shall be positioned at BOP 0.

During the format operation, the device server shall respond to commands as follows:

a) in response to all commands except REQUEST SENSE and INQUIRY, the device server shall return CHECK CONDITION unless a reservation conflict exists. In that case RESERVATION CONFLICT status shall be returned; or
b) in response to the REQUEST SENSE command, assuming no error has occurred, the device server shall return a sense key of NOT READY and the additional sense code shall be set to LOGICAL UNIT NOT READY, FORMAT IN PROGRESS, with the sense key specific bytes set for process indication (see SPC-4).

An immediate (IMMED) bit of zero specifies the device server shall not return status until the FORMAT MEDIUM command has completed. An IMMED bit of one specifies the device server shall return status as soon as the valid medium location has been verified and the command descriptor block of the FORMAT MEDIUM command has been validated. If CHECK CONDITION status is returned for a FORMAT MEDIUM command with an IMMED bit of one, the format operation shall not be performed.

A VERIFY bit of one specifies the logical unit shall format the medium and then verify that the format was successfully accomplished. The method used to verify success of the FORMAT MEDIUM command is vendor specific. If the verify operation determines that the format was not successfully accomplished, the device server shall return a sense key of MEDIUM ERROR and the additional sense code shall be set to MEDIUM FORMAT CORRUPTED. If the VERIFY bit is zero, the logical unit shall not perform the verify check.

The FORMAT field is defined in table 40.

Table 40 — Format field definition

Value	Description	Support
0h	Use default format	O
1h	Partition medium	O
2h	Default format then partition	O
3h-7h	Reserved	
8h-Fh	Vendor specific	

If the FORMAT field is 0h, the logical unit shall determine the format method to use. A valid FORMAT MEDIUM command with 0h in the FORMAT field shall cause all data on the entire physical volume to be lost.

If the FORMAT field is 1h, the logical unit shall partition the medium using the current mode data from the Medium Partition mode page (see 8.3.4). If none of the mode bits SDP, FDP, or IDP are set to one, the device server shall return CHECK CONDITION. The sense key shall be set to ILLEGAL REQUEST with the addition sense code set to PARAMETER VALUE INVALID. If insufficient space exists on the medium for the requested partition sizes, the device server shall return CHECK CONDITION status. The sense key shall be set to MEDIUM ERROR and the additional sense code shall be set to VOLUME OVERFLOW. A valid FORMAT MEDIUM command with 1h in the FORMAT field may cause all data on the entire physical volume to be lost.

If the FORMAT field is 2h, the logical unit shall perform the operations equivalent to a FORMAT field of 0h followed by a FORMAT field of 1h. A valid FORMAT MEDIUM command with 2h in the FORMAT field may cause all data on the entire physical volume to be lost.

When the FORMAT field contains 1h or 2h, some errors related to mode page field contents may not be detected until the FORMAT MEDIUM command is processed. Therefore, some error conditions described in 8.3.4 may be returned in response to a FORMAT MEDIUM command with 1h or 2h in the FORMAT field.

The TRANSFER LENGTH specifies the length in bytes of format information that shall transferred from the initiator. A transfer length of zero specifies no format information shall be transferred. This condition shall not be considered an error. If the FORMAT field is 0h, 1h, or 2h, the TRANSFER LENGTH shall be zero. Use of format information is restricted to vendor-specific values in the FORMAT field and the contents of the format information is vendor specific.

7.2 LOAD UNLOAD command

The LOAD UNLOAD command (see table 41) requests that the logical unit enable or disable the logical unit for further operations. This command may also be used to request a retension function. Prior to performing the LOAD UNLOAD operation, the device server shall perform a synchronize operation (see 4.2.10). If the buffered mode is not 0h (see 8.3) and a previous command was terminated with CHECK CONDITION status and the device is

unable to continue successfully writing, the logical unit shall discard any unwritten buffered logical objects prior to performing the LOAD UNLOAD operation.

Table 41 — LOAD UNLOAD command

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (1Bh)							
1	Reserved							IMMED
2	Reserved							
3	Reserved							
4	Reserved				HOLD	EOT	RETEN	LOAD
5	CONTROL							

An immediate (IMMED) bit of zero specifies the device server shall not return status until the load or unload operation has completed. If the IMMED bit is one, the device server shall return status as soon as all buffered logical objects have been written to the medium and the command descriptor block of the LOAD UNLOAD command has been validated. If CHECK CONDITION status is returned for a LOAD UNLOAD command with an IMMED bit of one, the load or unload operation shall not be performed.

NOTE 38 For compatibility with devices implemented prior to this version of the standard, a WRITE FILEMARKS command with an IMMED bit of zero should be used to perform a synchronize operation (see 4.2.10) prior to issuing a LOAD UNLOAD command with an IMMED bit of one.

A LOAD bit of one and a HOLD bit of zero specifies the medium in the logical unit shall be loaded and positioned to the beginning-of-partition zero. A LOAD bit of zero and a HOLD bit of zero specifies the medium in the logical unit shall be positioned for removal at the extreme position along the medium specified by the EOT bit. Following successful completion of an unload operation, the device server shall return CHECK CONDITION status with the sense key set to NOT READY for all subsequent medium-access commands until a new volume is mounted or a load operation is successfully completed.

A LOAD bit of one and a HOLD bit of one specifies if the medium has not been moved into the logical unit, the medium shall be moved in but not positioned for access. The EOT and RETEN bits shall be set to zero. Following successful completion, the device server shall return GOOD status. If both the medium and device server support MAM, the device server shall generate a unit attention condition for all initiators and the additional sense code shall be set to MEDIUM AUXILIARY MEMORY ACCESSIBLE.

A LOAD bit of zero and a HOLD bit of one specifies if the medium is in the logical unit, the medium shall be positioned as specified by the RETEN and EOT bits or shall be unthreaded (whichever is appropriate for the medium type) but shall not be demounted. Following successful completion, the device server shall return GOOD status.

A retension (RETEN) bit of one specifies the logical unit shall perform a retension function on the current medium. A RETEN bit of zero specifies the logical unit shall not perform a retension function on the current medium. Implementation of the retension function is vendor specific.

An end-of-tape (EOT) bit of one specifies an unload operation (LOAD bit set to zero) shall position the medium at end-of-medium for removal from the device. For devices that do not support unloading at end-of-medium the device server shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB. An EOT bit of zero specifies an unload operation shall position the medium at beginning-of-medium for removal from the device.

An EOT bit of one and a LOAD bit of one shall cause the device server to return CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB.

A HOLD bit of one specifies MAM shall be accessible upon completion of the command but the medium shall not be positioned for access. A HOLD bit of zero and a LOAD bit of one specifies the medium shall be positioned for access. A HOLD bit of zero and a LOAD bit of zero specifies MAM shall not be accessible upon completion of the command.

When operating in buffered mode 1h or 2h (see 8.3), the logical unit shall discard any unwritten buffered data after the LOAD UNLOAD command is validated if the device is unable to continue successfully writing (e.g., the device reported CHECK CONDITION status to a previous command, reported a write-type error, and the error has not been cleared or otherwise recovered).

7.3 LOCATE(16) command

The LOCATE(16) command (see table 42) causes the logical unit to position the medium to the logical object or logical file, as specified by the DEST_TYPE and LOGICAL IDENTIFIER fields. Upon completion, the logical position shall be as specified in table 43. Prior to performing the locate operation, the device server shall perform a synchronize operation (see 4.2.10).

Table 42 — LOCATE(16) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (92h)							
1	Reserved			DEST_TYPE		Rsvd	CP	IMMED
2	Reserved							BAM
3	PARTITION							
4	(MSB)							
5								
6								
7				LOGICAL IDENTIFIER				
8								
9								
10								
11								(LSB)
12	Reserved							
13	Reserved							
14	Reserved							
15	CONTROL							

An immediate (IMMED) bit of zero specifies the device server shall not return status until the locate operation has completed. If the IMMED bit is one, the device server shall return status as soon as all buffered logical objects have been written to the medium and the command descriptor block of the LOCATE(16) command has been validated. If

CHECK CONDITION status is returned for a LOCATE(16) command with an IMMED bit of one, the locate operation shall not be performed.

A change partition (CP) bit of one specifies a change to the partition specified in the PARTITION field shall occur prior to positioning to the logical object or logical file, as specified in the LOGICAL IDENTIFIER field. A CP bit of zero specifies no partition change shall occur and the PARTITION field shall be ignored.

The DEST_TYPE field shall be used in conjunction with the LOGICAL IDENTIFIER field to locate to the appropriate position of the medium. The DEST_TYPE field specifies whether the location specified is a logical object identifier or logical file identifier. The DEST_TYPE field is defined in table 43.

Table 43 — DEST_TYPE field definitions

Code	Description	Logical position upon successful completion	Support
00b	Logical object identifier	BOP side	M
01b	Logical file identifier	BOP side of the logical file	M
10b	Obsolete		
11b	Reserved		

A block address mode type (BAM) bit of zero specifies the logical unit shall process this command as an implicit address command. A BAM bit of one specifies the logical unit shall process this command as an explicit address command.

The LOGICAL IDENTIFIER field specifies the logical identifier to which the logical unit shall position the medium based on the current setting of the DEST_TYPE field. An otherwise valid LOCATE(16) command to any position between beginning-of-data and the position immediately after the last object in the partition (position at end-of-data) shall not return a sense key of ILLEGAL REQUEST. A LOCATE(16) to a position past end-of-data shall return CHECK CONDITION status and the sense key shall be set to BLANK CHECK. Additionally, the sense data EOM bit shall be set to one if end-of-data is located at or after early-warning.

If end-of-partition is encountered, CHECK CONDITION status shall be returned, the sense key shall be set to MEDIUM ERROR, and the sense data EOM bit shall be set to one.

The PARTITION field specifies the partition to select if the CP bit is one. Refer to the sequential-access device model (see 4.2.6) and the Medium Partition mode page (see 8.3.4) for additional information about partitioning.

The logical unit position is undefined if a LOCATE(16) command fails with a sense key other than ILLEGAL REQUEST.

7.4 PREVENT ALLOW MEDIUM REMOVAL command

The PREVENT ALLOW MEDIUM REMOVAL command (see table 44) requests that the logical unit enable or disable the removal of the medium. The logical unit shall not allow medium removal if any initiator port currently has medium removal prevented.

Table 44 — PREVENT ALLOW MEDIUM REMOVAL command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (1Eh)							
1	Reserved							
2	Reserved							
3	Reserved							
4	Reserved							PREVENT
5	CONTROL							

Table 45 specifies the PREVENT field values and their meanings.

Table 45 — PREVENT field

PREVENT	Description
00b	Medium removal shall be allowed.
01b	Medium removal shall be prohibited.
10b	Obsolete
11b	Obsolete

The prevention of medium removal shall begin when any application client issues a PREVENT ALLOW MEDIUM REMOVAL command with a PREVENT field of 01b (i.e., medium removal prevented). The prevention of medium removal for the logical unit shall terminate after:

a) one of the following occurs for each I_T nexus that previously had medium removal prevented:
   A) receipt of a PREVENT ALLOW MEDIUM REMOVAL command with a PREVENT field of 00b;
   B) an I_T nexus loss; or
b) a power on;
c) a hard reset; or
d) a logical unit reset.

~~If possible, the device server shall perform an synchronize cache operation before terminating the prevention of medium removal.~~ If a persistent reservation or registration is being preempted by a PERSISTENT RESERVE OUT command with PREEMPT AND ABORT service action (see SPC-4), the equivalent of a PREVENT ALLOW MEDIUM REMOVAL command with the PREVENT field set to zero shall be processed for each the I_T nexuses associated with the persistent reservation or registrations being preempted. This allows an application client to override the prevention of medium removal function for an initiator port that is no longer operating correctly.

While a prevention of medium removal condition is in effect, the logical unit shall inhibit mechanisms that normally allow removal of the medium by an operator.

7.5 READ BLOCK LIMITS command

The READ BLOCK LIMITS command (see table 46) requests that the READ BLOCK LIMITS data (see table 47) be returned. The READ BLOCK LIMITS data (see table 47) specifies the block length limits capability of the logical unit.

Table 46 — READ BLOCK LIMITS command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (05h)							
1	Reserved							
2	Reserved							
3	Reserved							
4	Reserved							
5	CONTROL							

Table 47 — READ BLOCK LIMITS data

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved			GRANULARITY				
1	(MSB)							
2				MAXIMUM BLOCK LENGTH LIMIT				
3								(LSB)
4	(MSB)							
5				MINIMUM BLOCK LENGTH LIMIT				(LSB)

The GRANULARITY field specifies the supported block size granularity. The logical unit shall support all block sizes *n* such that *n* minus the minimum block length limit is a multiple of $2^{GRANULARITY}$ and *n* is greater than or equal to the MINIMUM BLOCK LENGTH LIMIT and less than or equal to the MAXIMUM BLOCK LENGTH LIMIT.

If the MAXIMUM BLOCK LENGTH LIMIT value equals the MINIMUM BLOCK LENGTH LIMIT value, the logical unit supports fixed-block transfers only, with the block length equal to the MINIMUM BLOCK LENGTH LIMIT value. In this case, READ and WRITE commands with the FIXED bit set to zero shall result in CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB.

For read and write commands with the FIXED bit set to one, block lengths are limited to multiples of four (see 8.3).

If the MAXIMUM BLOCK LENGTH LIMIT value is not equal to the MINIMUM BLOCK LENGTH LIMIT value, the logical unit supports fixed-block transfers or variable-block transfers, with the block length constrained between the given limits in either transfer mode. The transfer mode is controlled by the FIXED bit in the WRITE or READ commands. If the maximum block limit is zero, a maximum block length is not specified.

7.6 READ POSITION command

7.6.1 READ POSITION command description

The READ POSITION command (see table 48) reports the current position and provides information about logical objects contained in the object buffer. No medium movement shall occur as a result of responding to the command.

Table 48 — READ POSITION command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (34h)							
1	Reserved			SERVICE ACTION				
2	Reserved							
3	Reserved							
4	Reserved							
5	Reserved							
6	Reserved							
7	(MSB)	ALLOCATION LENGTH						
8								(LSB)
9	CONTROL							

The service actions defined for the READ POSITION command are shown in table 49.

Table 49 — READ POSITION service action codes

Code	Name	Description	Implementation Requirements	Reference
00h	SHORT FORM -- BLOCK ID	Device server shall return 20 bytes of data with the FIRST LOGICAL OBJECT LOCATION and LAST LOGICAL OBJECT LOCATION fields as logical object identifier values (see 4.2.7.2), relative to a partition.	Mandatory	7.6.2
01h	SHORT FORM -- VENDOR SPECIFIC	Device server shall return 20 bytes of data with the FIRST LOGICAL OBJECT LOCATION and LAST LOGICAL OBJECT LOCATION fields as vendor-specific values.	Optional	7.6.2
02h	Reserved	Illegal request		
03h	Reserved	Illegal request		
04h	Reserved	Illegal request		
05h	Reserved	Illegal request		
06h	LONG FORM	Device server shall return 32 bytes of data.	Mandatory	7.6.3
07h	Reserved	Illegal request		

Table 49 — READ POSITION service action codes (Continued)

Code	Name	Description	Implementation Requirements	Reference
08h	EXTENDED FORM	Device server shall return 28 bytes of data up to the maximum length specified by the ALLOCATION LENGTH field.	Optional	7.6.4
09h - 1Fh	Reserved	Illegal request		

If the device server does not implement the specified service action code, then the command shall be terminated with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to INVALID FIELD IN CDB.

To maintain compatibility with earlier implementations, the service action codes 02h, 03h, 04h, 05h, and 07h shall not be implemented.

For service action codes of 00h, 01h, and 06h, the ALLOCATION LENGTH field in the CDB shall be zero. If it is not, then the command shall be terminated with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to INVALID FIELD IN CDB.

For a service action code of 08h, the ALLOCATION LENGTH field in the CDB specifies how much space has been allocated for the parameter data. If the length is not sufficient to contain the parameter data, the first portion of the parameter data shall be returned. This shall not be considered an error. If the remainder of the parameter data is required, the application client should send a new READ POSITION command with an ALLOCATION LENGTH field large enough to contain all the parameter data.

7.6.2 READ POSITION data format, short form

Table 50 specifies the READ POSITION data that shall be returned if the SERVICE ACTION field is 00h or 01h.

Table 50 — READ POSITION data format, short form

Bit / Byte	7	6	5	4	3	2	1	0
0	BOP	EOP	LOCU	BYCU	Rsvd	LOLU	PERR	BPEW
1	PARTITION NUMBER							
2	Reserved							
3	Reserved							
4	(MSB)							
5				FIRST LOGICAL OBJECT LOCATION				
6								
7								(LSB)
8	(MSB)							
9				LAST LOGICAL OBJECT LOCATION				
10								
11								(LSB)
12	Reserved							
13	(MSB)							
14				NUMBER OF LOGICAL OBJECTS IN OBJECT BUFFER				
15								(LSB)
16	(MSB)							
17				NUMBER OF BYTES IN OBJECT BUFFER				
18								
19								(LSB)

A beginning–of-partition (BOP) bit of one specifies the logical unit is at the beginning-of-partition in the current partition. A BOP bit of zero specifies the current logical position is not at the beginning-of-partition.

An end-of-partition (EOP) bit of one specifies the logical unit is positioned between early-warning and end-of-partition in the current partition. An EOP bit of zero specifies the current logical position is not between early-warning and end-of-partition.

A logical object count unknown (LOCU) bit of one specifies the NUMBER OF LOGICAL OBJECTS IN OBJECT BUFFER field does not represent the actual number of logical objects in the object buffer. A LOCU bit of zero specifies the NUMBER OF LOGICAL OBJECTS IN OBJECT BUFFER field is valid.

A byte count unknown (BYCU) bit of one specifies the NUMBER OF BYTES IN OBJECT BUFFER field does not represent the actual number of bytes in the object buffer. A BYCU bit of zero specifies the NUMBER OF BYTES IN OBJECT BUFFER field is valid.

A logical object location unknown (LOLU) bit of one specifies the first logical object location, last logical object location, or partition number are not currently known or not otherwise obtainable. A LOLU bit of zero specifies the FIRST LOGICAL OBJECT LOCATION, LAST LOGICAL OBJECT LOCATION, and PARTITION NUMBER fields contain valid position information.

A position error (PERR) bit of one specifies the logical unit is unable to report the correct position due to an overflow of any of the returned position data. A PERR bit of zero specifies an overflow has not occurred in any of the returned position data fields.

If the Beyond Programmable Early Warning (BPEW) bit is set to one, then the logical object location is in a PEWZ or on the EOP side of EW. If the BPEW bit is set to zero, then the logical object location is not in a PEWZ or on the EOP side of EW. The BPEW bit shall be set to zero if the LOLU bit is set to one or if the PEWS field (see 8.3.8) is set to zero.

The PARTITION NUMBER field reports the partition number for the current logical position. If the logical unit only supports one partition for the medium, the PARTITION NUMBER field shall be set to zero.

The FIRST LOGICAL OBJECT LOCATION field specifies the logical object identifier associated with the current logical position. The value shall specify the logical object identifier of the next logical object to be transferred between an application client and the device server if a READ or WRITE command is issued.

The LAST LOGICAL OBJECT LOCATION field specifies the logical object identifier associated with the next logical object to be transferred from the object buffer to the medium. If the object buffer does not contain a complete logical object or is empty, the value reported for the last logical object location shall be equal to the value reported for the first logical object location.

NOTE 39 The information provided by the FIRST LOGICAL OBJECT LOCATION and LAST LOGICAL OBJECT LOCATION fields may be used in conjunction with the LOCATE command to position the medium at the appropriate logical object on another device in the case of unrecoverable errors on the first device.

The NUMBER OF LOGICAL OBJECTS IN OBJECT BUFFER field specifies the number of logical objects in the object buffer of the logical unit that have not been written to the medium.

The NUMBER OF BYTES IN OBJECT BUFFER field specifies the total number of data bytes in the object buffer of the logical unit that have not been written to the medium.

7.6.3 READ POSITION data format, long form

Table 51 specifies the format of the READ POSITION data that shall be returned if the SERVICE ACTION field is 06h.

Table 51 — READ POSITION data format, long form

Bit / Byte	7	6	5	4	3	2	1	0
0	BOP	EOP	Reserved		MPU	LONU	Rsvd	BPEW
1	Reserved							
2	Reserved							
3	Reserved							
4	(MSB)	PARTITION NUMBER						
5								
6								
7								(LSB)
8	(MSB)	LOGICAL OBJECT NUMBER						
9								
10								
11								
12								
13								
14								
15								(LSB)
16	(MSB)	LOGICAL FILE IDENTIFIER						
17								
18								
19								
20								
21								
22								
23								(LSB)
24	(MSB)	Obsolete						
25								
26								
27								
28								
29								
30								
31								(LSB)

The BOP, EOP, and PARTITION NUMBER fields are as defined in the READ POSITION data format, short form (see table 50).

A mark position unknown (MPU) bit of one specifies the logical file identifier is not known or accurate reporting is not currently available. A MPU bit of zero specifies the LOGICAL FILE IDENTIFIER field contains valid position information.

A logical object number unknown (LONU) bit of one specifies the logical object number or partition number are not known or accurate reporting is not currently available. A LONU bit of zero specifies the LOGICAL OBJECT NUMBER and PARTITION NUMBER fields contain valid position information.

If the Beyond Programmable Early Warning (BPEW) bit is set to one, then the logical object location is in a PEWZ or on the EOP side of EW. If the BPEW bit is set to zero, then the logical object location is not in a PEWZ or on the EOP side of EW. The BPEW bit shall be set to zero if the LONU bit is set to one or if the PEWS field (see 8.3.8) is set to zero.

The PARTITION NUMBER field reports the partition number for the current logical position. If the logical unit only supports one partition for the medium, the PARTITION NUMBER field shall be set to zero.

The LOGICAL OBJECT NUMBER field specfies the number of logical objects between beginning-of-partition and the current logical position. A filemark counts as one logical object.

The LOGICAL FILE IDENTIFIER field specifies the number of filemarks between beginning-of-partition and the current logical position. This value is the current logical file identifier.

7.6.4 READ POSITION data format, extended form

Table 52 specifies the format of the READ POSITION data that shall be returned if the SERVICE ACTION field is 08h.

Table 52 — READ POSITION data format, extended form

Bit / Byte	7	6	5	4	3	2	1	0
0	BOP	EOP	LOCU	BYCU	Rsvd	LOLU	PERR	BPEW
1	PARTITION NUMBER							
2	(MSB)							
3	ADDITIONAL LENGTH (1Ch)							(LSB)
4	Reserved							
5	(MSB)							
6	NUMBER OF LOGICAL OBJECTS IN OBJECT BUFFER							
7								(LSB)
8	(MSB)							
9								
10								
11								
12	FIRST LOGICAL OBJECT LOCATION							
13								
14								
15								(LSB)
16	(MSB)							
17								
18								
19								
20	LAST LOGICAL OBJECT LOCATION							
21								
22								
23								(LSB)
24	(MSB)							
25								
26								
27								
28	NUMBER OF BYTES IN OBJECT BUFFER							
29								
30								
31								(LSB)

The fields are defined the same as for the corresponding fields in the READ POSITION data format, short form (see table 50).

The ADDITIONAL LENGTH field shall contain 1Ch. If the information transferred to the Data-In Buffer is truncated because of an insufficient ALLOCATION LENGTH value, the ADDITIONAL LENGTH field shall not be altered to reflect the truncation.

7.7 RECOVER BUFFERED DATA command

The RECOVER BUFFERED DATA command (see table 53) is used to recover data that has been transferred to the logical unit's object buffer but has not been successfully written to the medium. It is normally used to recover the buffered data after error or exception conditions make it impossible to write the buffered data to the medium. One or more RECOVER BUFFERED DATA commands may be required to recover all unwritten buffered data.

Table 53 — RECOVER BUFFERED DATA command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (14h)							
1	Reserved						SILI	FIXED
2	(MSB)							
3				TRANSFER LENGTH				
4								(LSB)
5	CONTROL							

The processing of this command is similar to the READ(6) command except that the data is transferred from the logical unit's object buffer instead of the medium. The order that logical block(s) are transferred is defined by the RBO bit in the Device Configuration mode page (see 8.3.3). If the RBO bit is not implemented, logical block(s) are transferred in the same order they would have been transferred to the medium.

Refer to the READ(6) command (see 6.4) for a description of the FIXED bit, the SILI bit, the TRANSFER LENGTH field, and any conditions that may result from incorrect usage of these fields.

If the FIXED bit is zero, no more than the requested transfer length shall be transferred to the application client. If the requested transfer length is smaller than the actual length of the logical block to be recovered, only the requested transfer length shall be transferred to the application client and the remaining data for the current logical block shall be discarded.

NOTE 40 During recovery operations involving unknown block sizes, the application client should select the maximum block length supported by the logical unit to ensure that all buffered data will be transferred and set the FIXED bit to zero.

If a buffered filemark is encountered during a RECOVER BUFFERED DATA command, CHECK CONDITION status shall be returned, the sense key shall be set to NO SENSE, and the FILEMARK and VALID bits shall be set to one in the sense data. Upon termination, the logical position shall be after the filemark. If the FIXED bit is one, the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks transferred. If the FIXED bit is zero, the INFORMATION field shall be set to the requested transfer length.

If an attempt is made to recover more logical blocks of data than are contained in the logical unit's object buffer, CHECK CONDITION status shall be returned, the sense key shall be set to NO SENSE. The additional sense code

shall be set to END-OF-DATA DETECTED, and the EOM and VALID bits shall be set to one in the sense data. If the FIXED bit is one, the INFORMATION field shall be set to the requested transfer length minus the actual number of logical blocks transferred. If the FIXED bit is zero, the INFORMATION field shall be set to the requested transfer length.

7.8 REPORT DENSITY SUPPORT command

7.8.1 REPORT DENSITY SUPPORT command description

The REPORT DENSITY SUPPORT command (see table 54) requests that information regarding the supported densities or medium type for the logical unit be sent to the application client.

Table 54 — REPORT DENSITY SUPPORT command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (44h)							
1	Reserved						MEDIUM TYPE	MEDIA
2	Reserved							
3	Reserved							
4	Reserved							
5	Reserved							
6	Reserved							
7	(MSB)	ALLOCATION LENGTH						
8								(LSB)
9	CONTROL							

If the MEDIA bit is set to zero, the device server shall return descriptors for densities or medium types supported by the logical unit for any supported media. If the MEDIA bit is set to one, the device server shall return descriptors for densities or the medium type supported by the mounted medium. If the MEDIA bit is set to one and the logical unit either contains no medium or contains a medium but cannot determine the medium's density or type, CHECK CONDITION status shall be returned. The sense key shall be set to NOT READY and the additional sense code shall specify the reason for NOT READY.

If the MEDIUM TYPE bit is set to zero, the device server shall return data as specified in 7.8.3. If the MEDIUM TYPE bit is set to one, the device server shall return data as specified in 7.8.4.

The ALLOCATION LENGTH field specifies the maximum number of bytes that the device server may return.

7.8.2 REPORT DENSITY SUPPORT header

The REPORT DENSITY SUPPORT header is specified in table 55.

Table 55 — REPORT DENSITY SUPPORT header

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1	AVAILABLE DENSITY SUPPORT LENGTH							(LSB)
2	Reserved							
3	Reserved							
4	Descriptor(s)							
n								

The AVAILABLE DENSITY SUPPORT LENGTH field specifies the number of bytes in the following data that is available to be transferred. The available density support length does not include itself. If the parameter data is truncated due to insufficient allocation length, the AVAILABLE DENSITY SUPPORT LENGTH field shall not be altered to reflect the truncation.

7.8.3 Density support report

The REPORT DENSITY SUPPORT command with a MEDIUM TYPE bit set to zero returns the REPORT DENSITY SUPPORT header (see table 55) followed by one or more density support data block descriptors (see table 56). The density support data block descriptors shall follow the REPORT DENSITY SUPPORT header. The density support data block descriptors shall be in numerical ascending order of the PRIMARY DENSITY CODE value.

Table 56 — Density support data block descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	PRIMARY DENSITY CODE							
1	SECONDARY DENSITY CODE							
2	WRTOK	DUP	DEFLT	Reserved				DLV
3	(MSB)							
4	DESCRIPTOR LENGTH							(LSB)
5	(MSB)							
6	BITS PER MM							
7								(LSB)
8	(MSB)							
9	MEDIA WIDTH							(LSB)
10	(MSB)							
11	TRACKS							(LSB)
12	(MSB)							
13	CAPACITY							
14								
15								(LSB)
16	ASSIGNING ORGANIZATION							
:								
23								
24	DENSITY NAME							
:								
31								
32	DESCRIPTION							
:								
51								

Density support data block descriptors shall be returned by ascending PRIMARY DENSITY CODE values. Multiple entries may exist for a given PRIMARY DENSITY CODE value. For all entries with equal PRIMARY DENSITY CODE values, all fields except for ASSIGNING ORGANIZATION, DENSITY NAME, and DESCRIPTION shall be identical. Density support data block descriptors with the same PRIMARY DENSITY CODE value should be ordered from most to least preferred ASSIGNING ORGANIZATION, DENSITY NAME, and DESCRIPTION.

NOTE 41 By allowing multiple entries for a given primary and secondary density code set, multiple standard names may identify the same density code. This facilitates the remapping of density codes, if required.

The density support data block descriptor may represent a particular format in addition to giving physical density information. The information in a density support data block descriptor provides an application client with a detailed review of the recording technologies supported by a logical unit. By supplying the density code value returned in a

density support data block descriptor in a MODE SELECT command (see 8.3), an application client selects the recording technology (density, format, etc.).

The PRIMARY DENSITY CODE field contains the value returned by a MODE SENSE command for the density described in the remainder of the density support data block descriptor. The device server shall accept a MODE SELECT command containing this value, for appropriate media. The value 7Fh shall be reserved. All other values are available for use. The value of 00h shall only be used for the default density of the logical unit.

When multiple density code values are assigned to the same recording technology (density, format, etc.), the SECONDARY DENSITY CODE field shall contain the equivalent density code value. When the SECONDARY DENSITY CODE is used in the mode select header with a MODE SELECT command, the device server shall accept this value as equivalent to the PRIMARY DENSITY CODE value. If no secondary density code exists, the device server shall return the PRIMARY DENSITY CODE value in this field.

A WRTOK bit of zero specifies the logical unit does not support writing to the media with this density. A WRTOK bit of one specifies the logical unit is capable of writing this density to either the currently mounted medium (MEDIA bit in command descriptor block set to one) or for some media (MEDIA bit in command descriptor block set to zero). All density code values returned by the REPORT DENSITY SUPPORT command shall be supported for read operations.

A DUP bit of zero specifies this primary density code has exactly one density support data block descriptor. A DUP bit of one specifies this primary density code is specified in more than one density support data block descriptor.

A DEFLT bit of zero specifies this density is not the default density of the drive. A DEFLT bit of one specifies this density is the default density. If either the PRIMARY DENSITY CODE or the SECONDARY DENSITY CODE field is zero, the DEFLT bit shall be one. If neither the primary or secondary density code is zero and the DEFLT bit is one, the logical unit shall accept a MODE SELECT header with a density code of 00h as equivalent to the primary and secondary density codes.

NOTE 42 The default density of the logical unit may vary depending on the currently mounted media. Multiple codes may return a DEFLT bit of one when the MEDIA bit is zero since more than one default may be possible.

If the Descriptor Length Valid (DLV) bit is set to one, the DESCRIPTOR LENGTH field shall contain the length of the descriptor minus 5. If the DLV bit is set to zero, the DESCRIPTOR LENGTH field shall be set to zero and the descriptor is 52 bytes in length.

The BITS PER MM field specifies the number of bits per millimeter per track as recorded on the medium. The value in this field shall be rounded up if the fractional value of the actual value is greater than or equal to 0,5. A value of 00h specifies the number of bits per millimeter does not apply to this logical unit. Direct comparison of this value between different vendors (possibly products) is discouraged since the definition of bits may vary.

The MEDIA WIDTH field specifies the width of the medium supported by this density. This field has units of tenths of millimeters. The value in this field shall be rounded up if the fractional value of the actual value is greater than or equal to 0,5. The MEDIA WIDTH field may vary for a given density depending on the mounted medium. A value of 00h specifies the width of the medium does not apply to this logical unit.

The TRACKS field specifies the number of data tracks supported on the medium by this density. The TRACKS value may vary for a given density depending on the mounted medium. Direct comparison of this value between different vendors (possibly products) is discouraged since the definition of the number of tracks may vary. For recording formats that are neither parallel nor serpentine, the TRACKS field specifies the maximum number of data tracks that are read or recorded simultaneously.

If the MEDIA bit is zero, the CAPACITY field specifies the approximate capacity of the longest supported medium assuming recording in this density with one partition. If the MEDIA bit is one, the CAPACITY field should specify the

approximate capacity of the current medium, assuming recording in this density with one partition. If the approximate capacity of the current medium is not available for the mounted medium, the longest supported medium capacity shall be used. If a SET CAPACITY command has affected the capacity of the medium, this shall be reflected in the CAPACITY field. The capacity assumes that compression is disabled, if possible. If this density does not support an uncompressed format, the capacity assumes that compression is enabled using average data. The capacity also assumes that the media is in good condition, and that normal data and block sizes are used. This value is in units of megabytes (10^6 bytes). The logical unit does not guarantee that this space is actually available in all cases. Direct comparison of this value between different vendors (possibly products) is discouraged since the length of media and the method used to measure maximum capacity may vary. The CAPACITY field is intended to be used by the application client to determine that the correct density is being used, particularly when a lower-density format is required for interchange.

The ASSIGNING ORGANIZATION field contains eight bytes of ASCII data identifying the organization responsible for the specifications defining the values in this density support data block descriptor. The data shall be left aligned within this field. The ASCII value for a space (20h) shall be used if padding is required. The ASSIGNING ORGANIZATION field should contain a value listed in the vendor identification list (see SPC-4). The use of a specific vendor identification, other than the one associated with the device is allowed.

NOTE 43 If vendor X defines a density and format, another vendor may use X in the ASSIGNING ORGANIZATION field. If exactly the same density and format construction later becomes known by another name, both X and the new assigning organization may be used for the density code. This is one condition that may result in multiple density support data block descriptors for a single density code value.

NOTE 44 It is intended that the ASSIGNING ORGANIZATION field contain a unique identification of the organization responsible for the information in a density support data block descriptor. In the absence of any formal registration procedure, T10 maintains a list of vendor and assigning organization identification codes in use. Vendors are requested to voluntarily submit their identification codes to prevent duplication of codes.

The DENSITY NAME field contains eight bytes of ASCII data identifying the document (or other identifying name) that is associated with this density support data block descriptor. The data shall be left aligned within this field. The ASCII value for a space (20h) shall be used if padding is required. Two physical densities (and possibly formats) shall not have identical ASSIGNING ORGANIZATION and DENSITY NAME fields. Assigning organizations are responsible for preventing duplicate usage of one density name for multiple different densities and/or formats.

NOTE 45 It is suggested that any document that specifies a format and density for the media contain the values to be used by a logical unit when reporting the density support. The values for the BITS PER MM, MEDIA WIDTH, and TRACKS should also be included in such a document to help maintain consistency.

The DESCRIPTION field contains twenty bytes of ASCII data describing the density. The data shall be left aligned within this field. The ASCII value for a space (20h) shall be used if padding is required.

7.8.4 Medium Type support report

The REPORT DENSITY SUPPORT command with a MEDIUM TYPE field bit set to one returns the REPORT DENSITY SUPPORT header (see table 55) followed by one or more medium type descriptors (see table 57). The

medium type descriptors shall follow the REPORT DENSITY SUPPORT header. The medium type descriptors shall be in numerical ascending order of the medium type value.

Table 57 — Medium type descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	MEDIUM TYPE							
1	Reserved							
2	(MSB)	DESCRIPTOR LENGTH (52)						
3								(LSB)
4	NUMBER OF DENSITY CODES							
5	(MSB)	PRIMARY DENSITY CODES						(LSB)
:	(MSB)							(LSB)
13	(MSB)							(LSB)
14	(MSB)	MEDIA WIDTH						
15								(LSB)
16	(MSB)	MEDIUM LENGTH						
17								(LSB)
18	Reserved							
19	Reserved							
20	ASSIGNING ORGANIZATION							
:								
27								
28	MEDIUM TYPE NAME							
:								
35								
36	DESCRIPTION							
:								
55								

The MEDIUM TYPE field contains the value returned by a MODE SENSE command for the medium type described in the remainder of the medium type descriptor. The device server shall accept a MODE SELECT command containing this value, for appropriate media.

The DESCRIPTOR LENGTH field contains the length of the descriptor minus 4.

The NUMBER OF DENSITY CODES field contains the number of valid density codes present in the PRIMARY DENSITY CODES value field.

The PRIMARY DENSITY CODES field contains a list of primary density code values supported by the drive for the medium type. The primary density code values shall be listed in ascending order. Any unused bytes in this field shall contain zero.

The MEDIA WIDTH field specifies the width of the medium. This field has units of tenths of millimeters. The value in this field shall be rounded up if the fractional portion of the actual value is greater than or equal to 0,5.

The MEDIUM LENGTH field specifies the nominal length of the medium. This field has units of meters. The value in this field shall be rounded up if the fractional portion of the actual value is greater than or equal to 0,5.

The ASSIGNING ORGANIZATION field contains eight bytes of ASCII data identifying the organization responsible for the specifications defining the values in this medium type descriptor. The data shall be left aligned within this field. The ASCII value for a space (20h) shall be used if padding is required. The ASSIGNING ORGANIZATION field should contain a value listed in the vendor identification list (see SPC-4). The use of a vendor identification other than the one associated with the device is allowed.

NOTE 46 It is intended that the ASSIGNING ORGANIZATION field contain a unique identification of the organization responsible for the information in a medium type descriptor. In the absence of any formal registration procedure, T10 maintains a list of vendor and assigning organization identification codes in use. Vendors are requested to voluntarily submit their identification codes to prevent duplication of codes.

The MEDIUM TYPE NAME field contains eight bytes of ASCII data identifying the document (or other identifying name) that is associated with this medium type descriptor. The data shall be left aligned within this field. The ASCII value for a space (20h) shall be used if padding is required. Two different medium types shall not have identical ASSIGNING ORGANIZATION and MEDIUM TYPE NAME fields. Assigning organizations are responsible for preventing duplicate usage of one medium type name for multiple different medium types.

NOTE 47 It is suggested that any document that specifies a characteristics for the media contain the values to be used by a logical unit when reporting the density support. The values for the MEDIUM WIDTH and MEDIUM LENGTH should also be included in such a document to help maintain consistency.

The DESCRIPTION field contains twenty bytes of ASCII data describing the medium type. The data shall be left aligned within this field. The ASCII value for a space (20h) shall be used if padding is required.

7.9 REWIND command

The REWIND command (see table 58) causes the logical unit to position to the beginning-of-partition in the current partition. Prior to performing the rewind operation, the device server shall perform a synchronize operation (see 4.2.10). If the buffered mode is not 0h (see 8.3) and a previous command was terminated with CHECK CONDITION status and the device is unable to continue successfully writing, the logical unit shall discard any unwritten buffered logical objects prior to performing the REWIND operation.

Table 58 — REWIND command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (01h)							
1	Reserved							IMMED
2	Reserved							
3	Reserved							
4	Reserved							
5	CONTROL							

An immediate (IMMED) bit of zero specifies the device server shall not return status until the rewind operation has completed. If the IMMED bit is one, the device server shall return status as soon as all buffered logical objects have been written to the medium and the command descriptor block of the REWIND command has been validated. If CHECK CONDITION status is returned for a REWIND command with an IMMED bit of one, the rewind operation shall not be performed.

NOTE 48 For compatibility with devices implemented prior to this standard, it is suggested that a WRITE FILEMARKS command with an IMMED bit of zero be used to perform a synchronize operation (see 4.2.10) before issuing a REWIND command with an IMMED bit of one.

7.10 SET CAPACITY command

The SET CAPACITY command (see table 59) sets the available medium for the currently mounted volume to a proportion of the total capacity of that volume. Any excess space shall be unavailable on the volume after successful completion of this command until changed by a new SET CAPACITY command. This change shall persist through power cycles, logical unit resets, I_T nexus losses, and unloading or reloading of the volume. Other vendor-specific actions such as physical erasure may change the total capacity of the volume. The method for recording the available capacity and other marks needed to manage the resulting capacity for volume interchange may be specified in a recording format standard or may be vendor specific.

Table 59 — SET CAPACITY command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (0Bh)							
1	Reserved							IMMED
2	Reserved							
3	(MSB)	CAPACITY PROPORTION VALUE						
4								(LSB)
5	CONTROL							

If the device server does not contain a medium, then the command shall be terminated with CHECK CONDITION status. The sense key shall be set to NOT READY, and the additional sense code shall be set to MEDIUM NOT PRESENT.

The SET CAPACITY command shall be accepted only when the medium is at beginning-of-partition 0 (BOP 0). If the medium is logically at any other position, the command shall be rejected with CHECK CONDITION status. The sense key shall be ILLEGAL REQUEST and the additional sense code shall be set to POSITION PAST BEGINNING OF MEDIUM.

A valid SET CAPACITY command shall cause all data and partitioning information on the entire physical volume to be lost. If the partitioning information changes, the device server shall generate a unit attention condition for all initiators with the additional sense code set to MODE PARAMETERS CHANGED.

An immediate (IMMED) bit of zero specifies the device server shall not return status until the set capacity operation has completed. An IMMED bit of one specifies the device server shall return status as soon as the command descriptor block of the SET CAPACITY command has been validated. If CHECK CONDITION status is returned for a SET CAPACITY command with an IMMED bit set to one, the set capacity operation shall not be performed.

The CAPACITY PROPORTION VALUE field specifies the portion of the total volume capacity to be made available for use. The CAPACITY PROPORTION VALUE field is the numerator to a fraction with a denominator of 65 535. The resulting available capacity on the volume shall be equal to the total volume capacity multiplied by this fraction. The device server may round up the capacity to the next highest supported value. This rounding shall not be considered an error and shall not be reported.

NOTE 49 Available and total volume capacities are approximate values that may be affected by defects that reduce the actual available capacity of the volume. Other factors, such as partitioning, compression, and logical block packing may also affect available capacity.

7.11 SPACE(16) command

The SPACE(16) command (see table 60) operates identically to the SPACE(6) command (see 6.6), but allows specifying a COUNT field up to eight bytes in length and has parameter data out that specifies the logical object identifier on the medium. Following completion of a SPACE(16) command a READ POSITION command should be issued to obtain positioning information.

Table 60 — SPACE(16) command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (91h)							
1	Reserved				CODE			
2	Reserved							
3	Reserved							
4	(MSB)							
5								
6								
7				COUNT				
8								
9								
10								
11								(LSB)
12	(MSB)							
13	PARAMETER LENGTH							(LSB)
14	Reserved							
15	CONTROL							

The CODE field is defined in table 35 (see 6.6).

When spacing over logical objects, the COUNT field specifies the number of logical objects to be spaced over in the current partition. A positive value *N* in the COUNT field when the CODE field is not 0011b (End-of-data) shall cause forward positioning (toward end-of-partition) over *N* logical objects ending on the end-of-partition side of the last logical object, if they exist. A zero value in the COUNT field when the CODE field is not 0011b (End-of-data) shall cause no change of logical position. A negative value -*N* (two's complement notation) in the COUNT field when the CODE field is not 0011b (End-of-data) shall cause reverse positioning (toward beginning-of-partition) over *N* logical

objects ending on the beginning-of-partition side of the last logical object, if they exist. When the CODE field is 0011b (End-of-data), the COUNT field shall be ignored and the device server shall perform a synchronize operation before moving before the end-of-data position. When the COUNT field is zero and the CODE field is not 0011b (End-of-data), a device server is not required to perform a synchronize operation. Support of spacing in the reverse direction is optional.

The PARAMETER LENGTH field is used to send parameter data space positioning information specifying the position on the medium from which to start the SPACE(16) command function. For an implicit block address mode command, the PARAMETER LENGTH field shall be set to 0. For an explicit block address mode command, the PARAMETER LENGTH field shall be set to 16. If the PARAMETER LENGTH field is set to any other value, the command shall be terminated with a CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN CDB.

Space positioning information is specified in table 61.

Table 61 — Space positioning information

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
1	Reserved							
2	Reserved							
3	PARTITION NUMBER							
4	(MSB)							
5								
6								
7								
8				LOGICAL OBJECT IDENTIFIER				
9								
10								
11								(LSB)
12	Reserved							
13	Reserved							
14	Reserved							
15	Reserved							

The LOGICAL OBJECT IDENTIFIER and PARTITION fields specify the position where the SPACE(16) command shall start. If the current logical position does not match the specified LOGICAL OBJECT IDENTIFIER and PARTITION fields, the device server shall perform a locate operation to the specified logical object identifier and partition prior to performing the space operation. If the locate operation fails, the device server shall return CHECK CONDITION status and the additional sense code shall be set to LOCATE OPERATION FAILURE. Following a locate operation failure the logical position is undefined.

NOTE 50 Locating to the logical object identifier prior to performing the space operation is necessary for the space operation to function properly when filemarks are between the starting logical object identifier and the expected ending logical object identifier of the space operation.

If a filemark is encountered while spacing over logical blocks, the command shall be terminated. CHECK CONDITION status shall be returned, and the FILEMARK and VALID bits shall be set to one in the sense data. The sense key shall be set to NO SENSE and the additional sense code shall be set to FILEMARK DETECTED. The INFORMATION field shall be set to the requested count minus the actual number of logical blocks spaced over. The logical position shall be on the end-of-partition side of the filemark if movement was in the forward direction and on the beginning-of-partition side of the filemark if movement was in the reverse direction.

NOTE 51 For some space operations using the SPACE(16) command, the INFORMATION field value may exceed the maximum value allowed in the fixed format sense data (see SPC-4). As such, the descriptor format sense data (see SPC-4) should be enabled (i.e., the D_SENSE bit is set to one in the Control mode page).

If early-warning is encountered while spacing over logical objects and the REW bit is set to one in the Device Configuration mode page (see 8.3.3), CHECK CONDITION status shall be returned, the sense key shall be set to NO SENSE, and the EOM and VALID bits shall be set to one in the sense data. The additional sense code shall be set to END-OF-PARTITION/MEDIUM DETECTED. The INFORMATION field shall be set to the requested count minus the actual number of logical objects spaced over as defined by the CODE value. If the REW bit is zero or the option is not supported by the logical unit, the device server shall not report CHECK CONDITION status at the early-warning point.

NOTE 52 Setting the REW bit to one is not recommended for most applications since data may be present after early-warning.

If end-of-data is encountered while spacing over logical objects, CHECK CONDITION status shall be returned, the sense key shall be set to BLANK CHECK, and the sense data VALID bit shall be set to one in the sense data. The additional sense code shall be set to END-OF-DATA DETECTED. The sense data EOM bit shall be set to one if end-of-data is encountered at or after early-warning. The INFORMATION field shall be set to the requested count minus the actual number of logical objects spaced over as defined by the CODE value. The medium shall be positioned such that a subsequent write operation would append to the last logical object.

If the end-of-partition is encountered while spacing forward over logical objects, CHECK CONDITION status shall be returned, and the sense key shall be set to MEDIUM ERROR. The additional sense code shall be set to END-OF-PARTITION/MEDIUM DETECTED, and the sense data EOM and VALID bit shall be set to one. The INFORMATION field shall be set to the requested count minus the actual number of logical objects spaced over as defined by the CODE value. The medium position following this condition is not defined.

If beginning-of-partition is encountered while spacing over logical objects in the reverse direction, the device server shall return CHECK CONDITION status and shall set the sense key to NO SENSE. The additional sense code shall be set to BEGINNING-OF-PARTITION/MEDIUM DETECTED. The sense data EOM and VALID bits shall be set to one, and the INFORMATION field set to the total number of logical objects not spaced over as defined by the CODE value (i.e., the requested number of logical objects minus the actual number of logical objects spaced over as defined by the CODE value). The medium position following this condition is not defined. A successfully completed SPACE(16) command shall not set EOM to one at beginning-of-partition.

When spacing over sequential filemarks, the count field is interpreted as follows:

a) a positive value *N* shall cause forward movement to the first occurrence of *N* or more consecutive filemarks being logically positioned after the N^{th} filemark;
b) a zero value shall cause no change in the logical position; or
c) a negative value *-N* (2's complement notation) shall cause reverse movement to the first occurrence of *N* or more consecutive filemarks being logically positioned on the beginning-of-partition side of the N^{th} filemark.

If end-of-partition is encountered while spacing to sequential filemarks, CHECK CONDITION status shall be returned, and the sense key shall be set to MEDIUM ERROR. The additional sense code shall be set to

END-OF-PARTITION/MEDIUM DETECTED, the EOM bit shall be set to one, and the VALID bit shall be set to zero in the sense data. The medium position following this condition is not defined.

If end-of-data is encountered while spacing to sequential filemarks, CHECK CONDITION status shall be returned, and the sense key shall be set to BLANK CHECK. The additional sense code shall be set to END-OF-DATA DETECTED, and the sense data VALID bit shall be set to zero. The medium shall be positioned such that a subsequent write operation would append to the last logical object. The sense data EOM bit shall be set to one if end-of-data is encountered at or after early-warning.

When spacing to end-of-data, the COUNT field is ignored. Upon successful completion, the medium shall be positioned such that a subsequent write operation would append to the last logical object.

If end-of-partition is encountered while spacing to end-of-data, CHECK CONDITION status shall be returned, and the sense key shall be set to MEDIUM ERROR. The additional sense code shall be set to END-OF-PARTITION/MEDIUM DETECTED, the EOM bit shall be set to one, and the VALID bit shall be set to zero in the sense data. The medium position following this condition is not defined.

8 Parameters for sequential-access devices

8.1 Diagnostic parameters

This subclause defines the descriptors and pages for diagnostic parameters used with sequential-access devices.

The diagnostic page codes for sequential-access devices are defined in table 62.

Table 62 — Diagnostic page codes

Page Code	Description	Reference
00h	Supported diagnostic pages	SPC-4
01h - 3Fh	Reserved (for all device types)	
40h - 7Fh	Reserved	
80h - FFh	Vendor specific	

8.2 Log parameters

8.2.1 Log parameters overview

This subclause defines the descriptors and pages for log parameters used with sequential-access devices.

The log page codes for sequential-access devices are defined in table 63.

Table 63 — Log page codes

Page Code	Description	Support	Reference
00h	Supported log pages	M	SPC-4
01h	Buffer Overrun/Underrun log page	O	SPC-4
02h	Write Error Counter log page	M	SPC-4
03h	Read Error Counter log page (read)	M	SPC-4
04h	Read Reverse Error Counter log page	M[a]	SPC-4
05h	Verify Error Counter log page	O	SPC-4
06h	Non-Medium Error log page	O	SPC-4
07h	Last *n* Error Events log page	O	SPC-4
08h - 0Ah	Reserved	-	
0Bh	Last *n* Deferred Error Events log page	O	SPC-4
0Ch	Sequential-Access Device log page	M	8.2.2
0Dh	Temperature log page	O	SPC-4
0Eh	Start-Stop Cycle Counter log page	O	SPC-4
0Fh	Application Client log page	O	SPC-4
a) Mandatory only if READ REVERSE command is supported.			

Table 63 — Log page codes (Continued)

Page Code	Description	Support	Reference
10h	Self-test Results log page	O	SPC-4
11h	DT Device Status log page	O	ADC-2
12h	TapeAlert Response log page	O	ADC-2
13h	Requested Recovery log page	O	8.2.7
14h	Device Statistics log page	O	8.2.4
15h	Reserved	-	
16h	Tape Diagnostic Data log page	O	8.2.5
17h-2Dh	Reserved	-	
2Dh	Current Service Information log page	O	8.2.6
2Eh	TapeAlert log page	M	8.2.3
2Fh	Informational Exceptions log page	O	SPC-4
30h - 3Fh	Vendor specific (does not require page format)	-	
a) Mandatory only if READ REVERSE command is supported.			

8.2.2 Sequential-Access Device log page

The Sequential-Access Device log page defines data counters associated with data bytes transferred to and from the medium and to and from the application client, binary list parameters describing native capacities, and a binary list parameter related to cleaning.

The default value for parameters 0 through 3 shall be zero.

NOTE 53 The data in parameters 0 and 1 are intended to provide an indication of the compression ratio for the written data. Parameters 2 and 3 are intended to provide an indication of the compression ratio for read data.

Table 64 defines the parameter codes for the Sequential-Access Device log page.

Table 64 — Parameter codes for Sequential-Access Device log page

Parameter Code	Description	Support
0000h	Number of data bytes received from application clients during WRITE command operations.	M
0001h	Number of data bytes written to the media as a result of WRITE command operations, not counting ECC and formatting overhead.	M
0002h	Number of data bytes read from the media during READ command operations, not counting ECC and formatting overhead.	M
0003h	Number of data bytes transferred to the initiator(s) during READ command operations.	M

Table 64 — Parameter codes for Sequential-Access Device log page (Continued)

Parameter Code	Description	Support
0004h	Approximate native capacity in megabytes (i.e., 10^6) from BOP to EOD. This is not sensitive to the current position of the medium. The approximate native capacity between EOD and EW is the difference of parameter 0005h and this parameter. There is no guarantee about the amount of data that can be written before reaching EW. A value of all bits set to one indicates that this information is invalid due to an unknown location of EOD (e.g., no medium is mounted, EOD information needs to be rebuilt).	M
0005h	Approximate native capacity in megabytes (i.e., 10^6) between BOP and EW of the current partition. If no volume is mounted the device server shall set all bits in this parameter to one.	M
0006h	Minimum native capacity in megabytes (i.e., 10^6) between EW and EOP of the current partition. This native capacity is assuming one-to-one compression (e.g., compression disabled), the medium is in good condition, and that the device recommended typical block size is used. If no volume is mounted the device server shall set all bits in this parameter to one.	M
0007h	Approximate native capacity in megabytes (i.e., 10^6) from BOP to the current position of the medium. If no volume is mounted the device server shall set all bits in this parameter to one.	M
0008h	Maximum native capacity in megabytes (i.e., 10^6) that is currently allowed to be in the device object buffer. This value may change depending on the current position of the medium (e.g., available native capacity may decrease as the current position of the medium approaches EOP).	M
0009h - 00FFh	Reserved.	-
0100h	Cleaning requested.	O
0101h - 7FFFh	Reserved.	-
8000h - FFFFh	Vendor-specific parameters.	-

NOTE 54 If the current partition has a native capacity of 200 GB (i.e. 200 * 10^9) with EW at 1GB prior to EOP and the medium is positioned at EOD which is at the point that is 75% of the native capacity between BOP and EW, then the device server would use the following to determine parameters 0004h, 0005h, and 0006h. Since 75% of native capacity is remaining, (200 GB - 1 GB) * 75% = 149,25 GB. This equation results in parameter 0004h = 149 250 (02 4702h), parameter 0005h = 199 000 (03 0958h), and parameter 0006h = 1 000 (00 03E8h).

A non-zero value of the cleaning requested parameter indicates that the device has requested a head cleaning and a subsequent cleaning cycle has not been completed. A zero value of the cleaning requested parameter indicates that the device has not requested a head cleaning. The cleaning requested parameter value shall persist across I_T nexus losses, logical unit resets, and power cycles.

8.2.3 TapeAlert log page

The TapeAlert log page (see table 65) defines error and informational flags used for detailed device diagnostics and management (see 4.2.17 and Annex A).

Table 65 — TapeAlert log page

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved		PAGE CODE (2Eh)					
1	Reserved							
2	(MSB)		PAGE LENGTH (n-3)					
3								(LSB)
	TapeAlert log parameter(s)							
4	TapeAlert log parameter (first) Length x = 5							
x+3								
n-y+1	TapeAlert log parameter (last) Length y = 5							
n								

See SPC-4 for a description of the PAGE CODE and PAGE LENGTH fields.

Table 66 specifies the format of a TapeAlert log parameter.

Table 66 — TapeAlert parameter format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)		PARAMETER CODE					
1								(LSB)
2	DU	DS (1)	TSD (1)	ETC	TMC		LBIN (0)	LP (0)
3	PARAMETER LENGTH (01h)							
4	Reserved							FLAG

The value in the PARAMETER CODE field shall range from 1 to 64.

See SPC-4 for a description of the DU bit, DS bit, TSD bit, ETC bit, TMC field, LBIN bit, and LP bit. The DS bit, TSD bit, LBIN bit, and LP bit shall be set to the value specified in table 66.

An active TapeAlert flag has the FLAG bit set to one. An inactive TapeAlert flag has the FLAG bit set to zero.

If processing a LOG SELECT command, the device server shall terminate the command with CHECK CONDITION status, set the sense key to ILLEGAL REQUEST, and set the additional sense code to INVALID FIELD IN PARAMETER LIST if the application client sends parameter data for the TapeALert log page with:

a) the TSD bit set to zero;
b) the DS bit set to zero;

c) the LP bit set to one;
d) the LBIN bit set to one;
e) the FLAG bit set to one; or
f) the PARAMETER LENGTH field set to a value other than 01h.

If the TASER bit is set to zero (see 8.3.8), the device server shall terminate the command with CHECK CONDITION stats, set the sense key to ILLEGAL REQUEST, and set the additional sense code to INVALID FIELD IN PARAMETER LIST upon processing a LOG SELECT command where the application client has sent parameter data for the TapeAlert log page with the ETC bit set to one.

8.2.4 Device Statistics log page

8.2.4.1 Device Statistics log page overview

The Device Statistics log page (see table 67) defines data counters associated with utilization of the tape device. A device server that implements the Device Statistics log page shall implement one or more of the defined parameters. Support for the individual parameters in the Device Statistics log page is optional. All supported parameters shall be persistent across I_T nexus loss, logical unit reset and power-on. The parameters shall not be set to zero or changed with the use of a LOG SELECT command.

Table 67 — Device Statistics log page

Bit / Byte	7	6	5	4	3	2	1	0
0	DS	SPF (0b)	PAGE CODE (14h)					
1	SUBPAGE CODE (00h)							
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
	Device Statistics log parameter(s)							
4	Device Statistics log parameter (first)							
	⋮							
n	Device Statistics log parameter (last)							

See SPC-4 for a description of the DS bit, SPF bit, PAGE CODE field, SUBPAGE CODE field and PAGE LENGTH field.

Table 68 specifies the Device Statistics log page parameter codes.

Table 68 — Device Statistics log parameter codes

Parameter Code	Description	Reference
0000h	Lifetime media loads	8.2.4.2
0001h	Lifetime cleaning operations	8.2.4.2
0002h	Lifetime power on hours	8.2.4.2
0003h	Lifetime media motion (i.e., head) hours	8.2.4.2
0004h	Lifetime meters of tape processed	8.2.4.2

Table 68 — Device Statistics log parameter codes (Continued)

Parameter Code	Description	Reference
0005h	Lifetime media motion (head) hours when incompatible media was last loaded	8.2.4.2
0006h	Lifetime power on hours when the last temperature condition occurred (i.e., TapeAlert code 24h)	8.2.4.2
0007h	Lifetime power on hours when the last power consumption condition occurred (i.e., TapeAlert code 1Ch)	8.2.4.2
0008h	Media motion (i.e., head) hours since last successful cleaning operation	8.2.4.2
0009h	Media motion (i.e., head) hours since second to last successful cleaning operation	8.2.4.2
000Ah	Media motion (i.e., head) hours since third to last successful cleaning operation	8.2.4.2
000Bh	Lifetime power on hours when the last operator initiated forced reset and/or emergency eject occurred	8.2.4.2
000Ch-0FFFh	Reserved	
1000h	Media motion (i.e., head) hours for each medium type	8.2.4.3
1001h-7FFFh	Reserved	
8000h-FFFFh	Vendor-specifc	

Parameter codes corresponding to values of time shall be reported in hours and rounded up to the next whole hour.

8.2.4.2 Device statistics data counter log parameter

The device statistics data counter log parameter format is specified in table 69.

Table 69 — Device statistics data counter log parameter format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1	PARAMETER CODE							(LSB)
2	DU	Obsolete	TSD (0b)	ETC	TMC		FORMAT AND LINKING (00b)	
3	PARAMETER LENGTH (n-3)							
4	(MSB)							
n	DEVICE STATISTICS DATA COUNTER							(LSB)

The PARAMETER CODE field is defined in table 68.

See SPC-4 for descriptions of the DU bit, TSD bit, ETC bit, TMC field and FORMAT AND LINKING field. The TSD bit and FORMAT AND LINKING field shall be set to the values specified in table 69.

The PARAMETER LENGTH field indicates the number of bytes in the DEVICE STATISTICS DATA COUNTER field that follows.

The DEVICE STATISTICS DATA COUNTER field is the value of the data counter associated with the parameter code.

8.2.4.3 Medium type log parameter

The medium type log parameter format is specified in table 70.

Table 70 — Medium type log parameter format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PARAMETER CODE (1000h)				(LSB)
2	DU (0b)	Obsolete	TSD (0b)	ETC (0b)	TMC (00b)		FORMAT AND LINKING (11b)	
3	PARAMETER LENGTH (n-3)							
	Medium type parameter(s)							
4	Medium type log parameter (first) (see table 71)							
	⋮							
n	Medium type log parameter (last) (see table 71)							

The PARAMETER CODE field shall be set to 1000h to indicate the Medium type log parameter.

See SPC-4 for descriptions of the DU bit, TSD bit, ETC bit, TMC field and FORMAT AND LINKING field. These fields shall be set to the values specified in table 70.

The PARAMETER LENGTH field indicates the number of bytes in the medium type parameters that follow.

The medium type parameter format is specified in table 71.

Table 71 — Medium type parameter format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved							
1	Reserved							
2	DENSITY CODE							
3	MEDIUM TYPE							
4	(MSB)							
7				MEDIA MOTION HOURS				(LSB)

The DENSITY CODE field contains the value returned in the general mode parameter block descriptor (see SPC-4).

The MEDIUM TYPE field contains the value returned in the mode parameter header (see SPC-4).

The value returned in the MEDIUM TYPE field is vendor specific for sequential-access devices.

The MEDIA MOTION HOURS field contains the number of media motion (i.e., head) hours for the type of medium specified by the combination of the MEDIUM TYPE field and DENSITY CODE field.

8.2.5 Tape Diagnostic Data log page

The Tape Diagnostic Data log page (see table 72) provides for a number of error-event records using the list parameter format. Each error-event record contains diagnostic information for a single error type encountered by the device including data counters associated with the error event, sense data, operation code/service action and medium type with associated media motion hours, etc. The Tape Diagnostic Data log page may be used to aid in field analysis and repair.

The Tape Diagnostic Data log page shall only include parameter entries for commands that terminated with a CHECK CONDITION status having the sense key set to MEDIUM ERROR, HARDWARE ERROR or ABORTED COMMAND.

The parameter code value associated with an error-event indicates the relative time at which a command terminated with a CHECK CONDITION status. A lower parameter code indicates that the command terminated with a CHECK CONDITION status at a more recent time. The parameter code values returned shall be numbered consecutively from 0000h (i.e., the most recent) up to n, where *n* is the number of current parameter entries. The number of supported parameter entries, *n,* is vendor specific.

In each parameter (see table 73) if the REPEAT bit is set to zero, then the parameter represents only one event. If the REPEAT bit is set to one, then the parameter represents more than one consecutive events that had the identical values for the MEDIUM ID NUMBER field, SENSE KEY field, ADDITIONAL SENSE CODE field and ADDITIONAL SENSE CODE QUALIFIER field in the parameter. If the REPEAT bit is set to one in the parameter, then other fields in the parameter shall be set to the values when the first of the consecutive events that had the identical values for the MEDIUM ID NUMBER field, SENSE KEY field, ADDITIONAL SENSE CODE field and ADDITIONAL SENSE CODE QUALIFIER field occurred.

All parameter codes shall be persistent across I_T nexus losses, logical unit resets, and power-on. The parameter entries shall not be set to zero or changed with the use of a LOG SELECT command.

Table 72 — Tape Diagnostic Data log page

Bit / Byte	7	6	5	4	3	2	1	0
0	DS	SPF (0b)	PAGE CODE (16h)					
1	SUBPAGE CODE (00h)							
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
Tape diagnostic data log parameter(s)								
4	Tape diagnostic data log parameter (first)							
	⋮							
n	Tape diagnostic data log parameter (last)							

See SPC-4 for a description of the DS bit, SPF bit, PAGE CODE field, SUBPAGE CODE field and PAGE LENGTH field.

The tape diagnostic data log parameter format is specified in table 73.

Table 73 — Tape diagnostic data log parameter format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)	PARAMETER CODE						
1								(LSB)
2	DU (0b)	Obsolete	TSD (0b)	ETC (0b)	TMC (00b)		FORMAT AND LINKING (11b)	
3	PARAMETER LENGTH (n-3)							
4	Reserved							
5	Reserved							
6	DENSITY CODE							
7	MEDIUM TYPE							
8	(MSB)	LIFETIME MEDIA MOTION HOURS						
11								(LSB)
12	Reserved							
13	REPEAT	Reserved			SENSE KEY			
14	ADDITIONAL SENSE CODE							
15	ADDITIONAL SENSE CODE QUALIFIER							
16	(MSB)	VENDOR-SPECIFIC CODE QUALIFIER						
19								(LSB)
20	(MSB)	PRODUCT REVISION LEVEL						
23								(LSB)
24	(MSB)	HOURS SINCE LAST CLEAN						
27								(LSB)
28	OPERATION CODE							
29	Reserved			SERVICE ACTION				
30	Reserved							
31	Reserved							

Table 73 — Tape diagnostic data log parameter format (Continued)

Bit / Byte	7	6	5	4	3	2	1	0
32	(MSB)							
:				MEDIUM ID NUMBER				
:								
63								(LSB)
64	Reserved						TIMESTAMP ORIGIN	
65	Reserved							
66	TIMESTAMP							
71								
72	VENDOR SPECIFIC							
n								

See SPC-4 for descriptions of the DU bit, TSD bit, ETC bit, TMC field and FORMAT AND LINKING field. These fields shall be set to the values specified in table 73.

The PARAMETER LENGTH field indicates the number of bytes in the tape diagnostic data log parameter data that follows.

The DENSITY CODE field contains the density code of the medium loaded at the time the command terminated with the CHECK CONDITION status. The DENSITY CODE field is the same value as returned in the general mode parameter block descriptor (see SPC-4). If no medium was loaded at the time the command terminated with the CHECK CONDITION status, then the DENSITY CODE field shall be set to 00h.

The MEDIUM TYPE field contains the type of medium loaded at the time the command terminated with the CHECK CONDITION status. The MEDIUM TYPE field is the same value as returned in the mode parameter header (see SPC-4). If no medium was loaded at the time the command terminated with the CHECK CONDITION status, then the MEDIUM TYPE field shall be set to 00h.

The LIFETIME MEDIA MOTION HOURS field contains the number of media motion (head) hours at the time the command terminated with the CHECK CONDITION status. The LIFETIME MEDIA MOTION HOURS field is equivalent to the value contained in the Device Statistics log page with a parameter code value of 0003h at the time the command terminated with the CHECK CONDITION status.

The REPEAT bit set to one indicates this parameter represents more than one consecutive events that had identical values for the MEDIUM ID NUMBER field, SENSE KEY field, ADDITIONAL SENSE CODE field, and ADDITIONAL SENSE CODE QUALIFIER field. The REPEAT bit set to zero indicates this parameter represents a single event.

See SPC-4 for descriptions of the SENSE KEY field, ADDITIONAL SENSE CODE field, and ADDITIONAL SENSE CODE QUALIFIER field. The SENSE KEY field, ADDITIONAL SENSE CODE field, and ADDITIONAL SENSE CODE QUALIFIER field shall contain the sense key and additional sense code values of the command that terminated with the CHECK CONDITION status.

The VENDOR-SPECIFIC CODE QUALIFIER field is vendor specific. The VENDOR-SPECIFIC CODE QUALIFIER may provide additional diagnostics information related to the command that terminated with the CHECK CONDITION status.

See SPC-4 for the descriptions of the PRODUCT REVISION LEVEL field. The PRODUCT REVISION LEVEL field shall contains the product revision level at the time the command terminated with the CHECK CONDITION status.

The HOURS SINCE LAST CLEAN field contains the time in media motion (i.e., head) hours since the last successful cleaning at the time the command terminated with the CHECK CONDITION status. The HOURS SINCE LAST CLEAN field is equivalent to the value contained in the Device Statistics log page with a parameter code of 0008h at the time the command terminated with the CHECK CONDITION status.

See SPC-4 for descriptions of the OPERATION CODE field and SERVICE ACTION field. The OPERATION CODE field and SERVICE ACTION field if applicable contain the operation code and service action of the command that terminated with the CHECK CONDITION status.

If medium was present at the time the command terminated with the CHECK CONDITION status, then the MEDIUM ID NUMBER field shall contain:

1) the BARCODE field value contained in the medium auxiliary memory (see SPC-4);
2) the MEDIUM SERIAL NUMBER field value contained in the medium auxiliary memory (see SPC-4); or
3) a vendor-specific value associated with the mounted medium.

If no medium was present at the time the command terminated with the CHECK CONDITION status, the MEDIUM ID NUMBER field shall be filled with 20h (i.e., ASCII space).

See SPC-4 for descriptions of the TIMESTAMP ORIGIN and TIMESTAMP fields. The TIMESTAMP ORIGIN field and TIMESTAMP field contain the timestamp origin and timestamp maintained by the device server at the time the command terminated with the CHECK CONDITION status. If a timestamp is not supported by the device server, the TIMESTAMP ORIGIN and TIMESTAMP fields shall be set to zero.

8.2.6 Current Service Information log page

8.2.6.1 Current Service Information log page overview

The Current Service Information log page (see table 74) specifies information used for detailed device diagnostics and management.

Table 74 — Current Service Information log page

Bit / Byte	7	6	5	4	3	2	1	0
0	DS	SPF (0b)	PAGE CODE (2Dh)					
1	SUBPAGE CODE (00h)							
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
	Service information log parameter(s)							
4	Service information log parameter (first)							
	⋮							
n	Service information log parameter (last)							

See SPC-4 for a description of the DS bit, SPF bit, PAGE CODE field, SUBPAGE CODE field and PAGE LENGTH field.

The service information log parameter format is specified in table 75.

Table 75 — Service information log parameter format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)	PARAMETER CODE						
1								(LSB)
2	DU	Obsolete	TSD	ETC (0b)	TMC (00b)		FORMAT AND LINKING (01b)	
3	PARAMETER LENGTH (x-3)							
4	Timestamp descriptor							
15								
16	Service information descriptor (first)							
r								
	⋮							
t	Service information descriptor (last)							
x								

See SPC-4 for descriptions of the DU bit, TSD bit, ETC bit, TMC field and FORMAT AND LINKING field. These fields shall be set to the values specified in table 75.

The PARAMETER LENGTH field indicates the number of bytes that follow.

The value in the PARAMETER CODE field shall be set to the Flag Number (see table 10) of the TapeAlert flag for which the information applies. When a TapeAlert flag is activated, the parameter in this log page relating to that TapeAlert flag is created. This parameter shall continue to be reported until overwritten by the next activation of the associated TapeAlert flag or until cleared by a Log Select command. The act of returning a parameter shall not clear that parameter and shall not cause deactivation of the TapeAlert flag.

The Timestamp descriptor is defined by the REPORT TIMESTAMP parameter data format (see SPC-4) with values reflecting the time the TapeAlert flag specified by the PARAMETER CODE field was activated.

Service information descriptors are returned and provide specific information about the TapeAlert flag. At least one service information descriptor shall be returned. The format of service information descriptors is specified in table 76.

Table 76 — Service information descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	SERVICE INFORMATION DESCRIPTOR TYPE							
1	SERVICE INFORMATION DESCRIPTOR LENGTH (n-1)							
2	Service information descriptor specific information (see table 77)							
n								

Only one service information descriptor shall be returned for a specific value of SERVICE INFORMATION DESCRIPTOR TYPE per parameter. The SERVICE INFORMATION DESCRIPTOR TYPE field is specified in table 77..

Table 77 — SERVICE INFORMATION DESCRIPTOR TYPE field

Code	Service Information Descriptor Type	Reference
00h	Vendor-specific service information	8.2.6.2
01h	Device information	8.2.6.3
02h	Volume information	8.2.6.4
03h	TapeAlert flag specific information	8.2.6.5
04h-FEh	Reserved	

8.2.6.2 Vendor-specific service information descriptor

Table 78 specifies the vendor-specific service information descriptor format.

Table 78 — Vendor-specific service information descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	SERVICE INFORMATION DESCRIPTOR TYPE (00h)							
1	SERVICE INFORMATION DESCRIPTOR LENGTH (n-1)							
2	Vendor-specific information							
n								

8.2.6.3 Device information descriptor

Table 79 specifies the device information descriptor format. The device information descriptor is returned when the cause of the TapeAlert flag relating to the parameter may be related to the device. There shall be only one device information descriptor returned per service information log parameter.

Table 79 — Device information descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	SERVICE INFORMATION DESCRIPTOR TYPE (01h)							
1	SERVICE INFORMATION DESCRIPTOR LENGTH (x-1)							
2	DEVICE SERVERITY CODE							
3	DEC							
4	DECQ							
5	DECT LENGTH							
6	DECT							
n								
n+1	NUMBER OF DEVICE REQUESTED RECOVERIES							
n+2	DEVICE REQUESTED RECOVERY (first)							
	⋮							
x	DEVICE REQUESTED RECOVERY (last)							

The DEVICE INFORMATION LENGTH field specifies the length of the information related to the device.

The DEVICE SEVERITY CODE field is specified in table 9.

The DEVICE ELEMENT CODE (DEC) field is specified in table 80.

Table 80 — DEVICE ELEMENT CODE field

Code	Description
00h	No message
10h	Device data path
20h	Mechanical
30h	Primary interface
40h	Automation interface
50h	Diagnostic interface
60h	Electronic elements
70h	Microcode
F0-FFh	Reserved

The DEVICE ELEMENT CODE QUALIFIER (DECQ) field is a vendor-specfic value providing more detailed information about the element specified by the DEC field.

The DECT LENGTH field specifies the length of the DECT field.

The DEVICE ELEMENT CODE TEXT (DECT) field is null-terminated and contains a description of what caused the TapeAlert flag to be activated.

The NUMBER OF DEVICE REQUESTED RECOVERIES field specifies the number of DEVICE REQUESTED RECOVERIES fields.

The DEVICE REQUESTED RECOVERY field values are defined in table 81 and shall be returned in prioritized order..

Table 81 — DEVICE REQUESTED RECOVERY field

Code	Description
00h	No recovery requested
01h	Retrieve device debug logs
02h	Clean device
03h	Update microcode
04h	Power off device and call service
05h	Leave the device in current state and call service
06h	Remove power from the device then apply power
07h-FFh	Reserved

8.2.6.4 Volume information descriptor

Table 82 specifies the volume information descriptor format. The volume information descriptor is returned when the cause of the TapeAlert flag relating to the parameter may be related to the volume.

Table 82 — Volume information descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	SERVICE INFORMATION DESCRIPTOR TYPE (02h)							
1	VOLUME INFORMATION LENGTH (n)							
2	VOLUME SERVERITY CODE							
3	VIC							
4	VICQ							
	Volume identification descriptor(s)							
5	VOLUME IDENTIFICATION LENGTH (n-5)							
6	Volume identification descriptor (first)							
x								
	⋮							
y	Volume identification descriptor (last)							
n								

The VOLUME INFORMATION LENGTH field specifies the length of the information related to the volume.

The VOLUME SEVERITY CODE field is specified in table 9.

The VOLUME INFORMATION CODE (VIC) field is specified in table 80.

Table 83 — VOLUME INFORMATION CODE field

Code	Description
00h	No message
01h	Good WORM volume
06h	Good encrypted volume
0Bh	Good data volume
10h	Good cleaning volume
15h	Good microcode update volume
1Ah	Bad WORM volume
1Fh	Bad encrypted volume
25h	Bad data volume
2Ah	Bad cleaning volume
2Fh	Bad microcode update volume
All others	Reserved

The VOLUME INFORMATION CODE QUALIFIER (VICQ) field is specified in table 84..

Table 84 — VOLUME INFORMATION CODE QUALIFIER field

Code	Description
00h	No message
01h	Read only permitted at this logical position
06h	Encryption key required
0Bh	Read only permitted for the entire volume
10h	Rewrite volume if possible
15h	Tape directory invalid, re-read volume if possible
1Ah	Cannot read or write
1Fh	Replace volume
25h	Auxiliary memory error
All others	Reserved

The VOLUME IDENTIFICATION LENGTH field specifies the length of the volume identification descriptors.

The volume identification descriptor format is the same as the MAM ATTRIBUTE format for medium auxiliary memory (see SPC-4). If the volume information descriptor is returned and:

1) a MAM attribute exsits for the volume identifier parameter of the device type attributes (i.e., set by the SMC device), then this attribute shall be returned as a volume identification descriptor;

2) a MAM attribute exists for the barcode parameter of the host type attributes (i.e., set by an application client), then this attribute shall be returned as a volume identification descriptor; and
3) a MAM attribute exists for the medium serial number parameter of the medium type attributes (i.e., set by the manufacturer), then this attribute shall be returned as a volume identification descriptor.

8.2.6.5 TapeAlert flag specific information

Table 85 specifies the TapeAlert flag information descriptor format. Table 10 specifies which flags are returned for this descriptor.

Table 85 — TapeAlert flag specific information descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	SERVICE INFORMATION DESCRIPTOR TYPE (03h)							
1	SERVICE INFORMATION DESCRIPTOR LENGTH (2)							
2	(MSB)							
n	CURRENT PERCENTAGE							(LSB)

The current percentage field specifies a signed percentage indicating how close to operating limits the item is. The value is the signed percentage multiplied by 16384. If the magnitude of the percentage is less than or equal to 100 percent, then the device is operating within specifications. If the magnitude is greater than 100 percent, then the device is outside the operating specifications. The equation that shall be used is:

$$\frac{measuredValue - \left[\frac{\langle upperLimit - lowerLimit \rangle}{2} + lowerLimit\right]}{upperLimit - \left[\frac{\langle upperLimit - lowerLimit \rangle}{2} + lowerLimit\right]} \times 16384$$

Example 1: if the power specification states the operation range is between 4.78 volts and 5.32 volts and the measured voltage is 4.70 volts, then the value returned by the equation is:

$$\frac{4.7 - \left[\frac{\langle 5.32 - 4.78 \rangle}{2} + 4.78\right]}{5.32 - \left[\frac{\langle 5.32 - 4.78 \rangle}{2} + 4.78\right]} \times 16384 = -21239 = AD09h$$

Example 2: if the media life is specified to be 260 full backups and the media has had 234 full backups performed, then the value returned by the equation is:

$$\frac{234 - \left[\frac{\langle 260 - 0 \rangle}{2} + 0\right]}{260 - \left[\frac{\langle 260 - 0 \rangle}{2} + 0\right]} \times 16384 = 13107 = 3333h$$

8.2.7 Requested Recovery log page

8.2.7.1 Requested Recovery log page overview

Table 86 specifies the Requested Recovery log page. If the device is unable to complete an action (e.g., a volume load or unload) the device server may set the rrqst bit to one in the very high frequency data log parameter (see ADC-2) to request that the application client perform a recovery action. The application client is able to obtain a list of alternative requested recovery actions by reading the Requested Recovery log page.

Table 86 — Requested Recovery log page

Bit / Byte	7	6	5	4	3	2	1	0
0	DS	SPF (0b)	PAGE CODE (13h)					
1	SUBPAGE CODE (00h)							
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
4		Reqested recovery log parameters						
n								

See SPC-4 for a description of the PAGE CODE field, the DS bit, then SPF bit, the SUBPAGE CODE field, and the PAGE LENGTH field.

Table 87 specifies the requested recovery log parameter codes.

Table 87 — Requested recovery log parameter codes

Parameter Code	Description	Reference
0000h	Recovery procedures	8.2.7.2
0001h-7FFFh	Reserved	
8000h-FFFFh	Vendor specific	

8.2.7.2 Recovery procedures log parameter

The recovery procedures log parameter format is specified in table 88.

Table 88 — Requested recovery log parameter format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1	PARAMETER CODE (0000h)							(LSB)
2	DU (1b)	Obsolete	TSD (1b)	ETC (0b)	TMC (00b)		FORMAT AND LINKING (11b)	
3	PARAMETER LENGTH (n-3)							
	Recovery procedures list							
4	Recovery procedure (first)							
	⋮							
n	Recovery procedure (last)							

See SPC-4 for descriptions of the DU bit, TSD bit, ETC bit, TMC field, and FORMAT AND LINKING field. These bits and fields shall be set to the values specified in table 88.

The PARAMETER LENGTH field specifies the number of recovery procedure bytes that follow.

The PARAMETER CODE field shall be set to 0000h to specify the reovery procedures log parameter.

The recovery procedures list contains recovery procedures (see table 89) listed in order from the most preferred to the least preferred procedure. If multiple recovery procedures are available, the most preferred procedure shall be the first in the list (i.e., in byte 4), and the other procedures listed in decreasing order of preference.

Each recovery procedure consists of one or more actions to be performed. If the INTXN bit in the VHF data descriptor of the DT Device Status log page (see ADC-2) is set to one, the parameter shall report only code 00h (i.e., Recovery not requested). If a failure occurs in performing one of the actions in a procedure, an appropriate list of requested recovery procedures may be reported.

Recovery procedures do not persist across a power cycle.

Table 89 — Recovery procedures

Recovery procedure	Description
00h	Recovery not requested.
01h	recovery requested, no recovery procedure defined.
02h	Instruct operator to push volume.
03h	Instruct operator to remove and re-insert volume.
04h	Issue UNLOAD command. Instruct operator to remove and re-insert volume.
05h	Instruct operator to power cycle target device.
06h	Issue LOAD command.

Table 89 — Recovery procedures

Recovery procedure	Description
07h	Issue UNLOAD command.
08h	Issue LOGICAL UNIT RESET task management function.
09h	No reovery procedure specified. Contact service organization.
0Ah	Issue UNLOAD command. Instruct operator to remove and quarantine volume.
0Bh	Instruct operator to not insert a volume. contact service organization.
0Ch	Issue UNLOAD command. Instruct operator to remove volume. Contact service organization.
0Dh	Request creation of a target device error log.
0Eh	Retrieve a target device error log.
0Fh	Modify configuration to allow microcode update and instruct operator to re-insert volume.
10h-7Fh	Reserved
80h-FFh	Vendor-specific procedures.

If the Requested Recovery log page is requested and the RRQST bit in the VHF data descriptor of the DT Device Status log page (see ADC-2) is set to zero, then a recovery procedure of 00h (i.e., Recovery not requested) shall be reported. If the requested recovery procedure is 09h (i.e., No recovery procedure defined. Contact service organization), then the application client shall not issue a load or unload command and the operator should not attempt to manipulate the volume physically.

If the requested recovery procedure is 0Ah (i.e., Issue UNLOAD command; Instruct operator to remove and quarantine volume), then the volume should be removed from service.

If the requested recovery procedure is 0Bh (i.e., Instruct operator to not insert a volume. Contact service organization), then a nonrecoverable error has occurred and insertion of a volume may cause damage. If the 0Bh recovery procedure is requested, then the RAA bit in the VHF data descriptor of the DT Device Status log page (see ADC-2) shall be set to zero, and no other recovery procedures shall be reported.

If the requested recovery procedure is 0Ch (i.e., Issue UNLOAD command. Instruct operator to remove volume; Contact service organization), then a non-recoverable error has occurred and insertion of a new volume may cause damage. If recovery procedure 0Ch is requested and the volume has been removed, then the RAA bit in the VHF data descriptor of the DT Device Status log page (see ADC-2) shall be set to zero, and no other recovery procedures shall be reported.

8.3 Mode parameters

8.3.1 Mode parameters overview

This subclause defines the descriptors and pages for mode parameters used with sequential-access devices.

The mode parameter list, including the mode parameter header and mode block descriptor, are described in SPC-4.

The MEDIUM TYPE field in the mode parameter header is vendor specific for sequential-access devices.

The value of the BLOCK LENGTH field in the mode parameter block descriptor shall be a multiple of four.

NOTE 55 The block length field is limited to multiples of four to ensure data integrity will be maintained when fixed-block transfers are performed using transports such as Fibre Channel.

The DEVICE-SPECIFIC PARAMETER field in the mode parameter header is defined in table 90 for sequential-access devices.

Table 90 — Device-specific parameter

Bit / Byte	7	6	5	4	3	2	1	0
	WP	BUFFERED MODE			SPEED			

When used with the MODE SENSE command, a write protection (WP) bit of zero specifies the medium is write enabled. A WP bit of one specifies the medium is currently in the write protected state. When used with the MODE SELECT command, this field is ignored.

NOTE 56 The write protected state may be due to logical unit internal restrictions, software write protection, or physical write protection.

Values for the BUFFERED MODE field are defined in table 91.

Table 91 — Buffered modes

Code	Description
0h	The device server shall not report GOOD status on WRITE commands until the logical blocks are actually written on the medium.
1h	The device server may report GOOD status on WRITE commands as soon as all the data specified in the WRITE command has been transferred to the logical unit's object buffer. One or more logical blocks may be buffered prior to writing the logical block(s) to the medium.
2h	The device server may report GOOD status on WRITE commands as soon as: a) all the data specified in the write command has been successfully transferred to the logical unit's object buffer; and b) all buffered logical objects from different initiators has been successfully written to the medium.
3h - 7h	Reserved

Values for the SPEED field shall be assigned as defined in table 92.

Table 92 — SPEED field definition

Code	Description
0h	Default (use the device's default speed).

Table 92 — SPEED field definition

Code	Description
1h	Use the device's lowest speed.
2h - Fh	Use increasing device speeds.

For the MODE SELECT command, the DENSITY CODE field of the sequential-access device block descriptor (see SPC-4) specifies the density selected by the application client for use in subsequent read and write operations. For logical units capable of automatic density recognition, the density code selected by the application client may be overridden by the logical unit for a subsequent read operation if the selected value does not match the current recorded density of the medium. If the MODE SELECT command specifies the default density code the logical unit selects the actual density code to be used in a vendor-specific manner. The value is expected to be the principal density code (or an optimal density code).

For the MODE SENSE command, the DENSITY CODE field reflects the current operating density of the logical unit. If a current operating density has not been selected, either because no medium is mounted or because the density of the installed medium has not been determined, the DENSITY CODE field should be set to the principal density code value (see 3.1.54). For some logical units, the principal density code value returned in response to a MODE SENSE command may change dynamically to match the most recently detected density. The DENSITY CODE value returned in response to a MODE SENSE command shall be determined as follows:

a) following a logical unit reset, if the logical unit is not ready, the device server shall report the principal density;
b) following a unit attention condition for a not-ready-to-ready transition or an unsuccessful read operation, the device server shall:
   A) report the principal density if no attempt has been made by the logical unit to determine the density;
   B) report the principal density if the logical unit is unable to automatically determine the density from the medium; or
   C) report the current medium density if the logical unit has determined the density from the medium.
c) following a successful read operation, the device server shall report a density code value reflecting the recorded density of the medium. For some implementations, the logical unit may automatically determine this value from the medium. For devices not capable of automatic density determination, the principal density is reported if the density code value is not provided by the preceding MODE SELECT command;
d) following a successful write operation, the device server shall report a density code value reflecting the most recently recorded density of the medium;
e) following an unsuccessful read operation or a successful write operation, while at beginning-of-partition, the device server shall report a density code value as described for item b);
f) following a successful unload operation, the device server shall report the most recent density code value as determined by items b) through e) above; or
g) following a logical unit reset, if the logical unit is ready, the device server shall retain knowledge of the density code as determined by items b) through e) above.

For a MODE SELECT command, a density code of 7Fh specifies the application client is not selecting a density. The value 7Fh shall not be returned by a MODE SENSE command. Table 93 lists the sequential-access device density codes.

Table 93 — Sequential-access density codes

Code value	Description	Note
00h	Default density.	a
01h - 7Eh	Density code from REPORT DENSITY SUPPORT command.	

Table 93 — Sequential-access density codes

Code value	Description	Note
7Fh	No change from previous density (NO-OP).	b
80h - FFh	Density code from REPORT DENSITY SUPPORT command.	

a. Only reported by MODE SENSE commands if primary density code for the density.
b. This density code value is defined for the MODE SELECT command and shall not be returned by the MODE SENSE command.

The mode page codes and subpage codes for sequential-access devices are defined in table 94. All page code and subpage code combinations not shown in table 94 are reserved.

Table 94 — Mode page codes and subpage codes

Page code	Subpage code	Mode page name	Support	Reference
00h	not applicable	Vendor specific (does not require page format)	-	
01h	00h	Read-Write Error Recovery	M	8.3.5
02h	00h	Disconnect-Reconnect	M	SPC-4
03h - 08h	00h - FEh	Reserved	-	
09h	00h	Obsolete	-	3.3.7
0Ah	00h	Control	M	SPC-4
0Ah	01h	Control Extension	O	SPC-4
0Bh - 0Eh	00h - FEh	Reserved	-	
0Fh	00h	Data Compression	M	8.3.2
10h	00h	Device Configuration	M	8.3.3
10h	01h	Device Configuration Extension	O	8.3.8
11h	00h	Medium Partition	O	8.3.4
12h	00h	Obsolete	-	3.3.7
13h	00h	Obsolete	-	3.3.7
14h	00h	Obsolete	-	3.3.7
15h	00h	Extended	O	SPC-4
16h	00h	Extended Device-Type Specific	-	SPC-4
17h	00h - FEh	Reserved	-	
18h	00h	Protocol Specific LUN	M[b]	SPC-4

a) Valid only for MODE SENSE command.
b) Mandatory only if explicit command set is supported.
c) Mandatory only if supported by the SCSI transport protocol.
d) See SPC-4 for support requirements.

Table 94 — Mode page codes and subpage codes

Page code	Subpage code	Mode page name	Support	Reference
19h	00h	Protocol Specific Port	M[c]	SPC-4
1Ah	00h	Power Condition	O	SPC-4
1Bh	00h - FEh	Reserved	-	
1Ch	00h	Informational Exceptions Control	M	8.3.6
1Dh	00h	Medium Configuration	M	8.3.7
1Dh - 1Fh	00h - FEh	Reserved	-	
20h - 3Eh	00h - FEh	Vendor specific (does not require page format)	-	
3Fh	00h	Return all pages[a]	_[d]	SPC-4
3Fh	FFh	Return all pages and subpages[a]	_[d]	SPC-4
00h - 3Eh	FFh	Return all subpages[a]	_[d]	SPC-4

a) Valid only for MODE SENSE command.
b) Mandatory only if explicit command set is supported.
c) Mandatory only if supported by the SCSI transport protocol.
d) See SPC-4 for support requirements.

8.3.2 Data Compression mode page

The Data Compression mode page (see table 95) specifies the parameters for the control of data compression in a sequential-access device.

Table 95 — Data Compression mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF(0)	PAGE CODE (0Fh)					
1	PAGE LENGTH (0Eh)							
2	DCE	DCC	Reserved					
3	DDE	RED		Reserved				
4	(MSB)							
5				COMPRESSION ALGORITHM				
6								
7								(LSB)
8	(MSB)							
9				DECOMPRESSION ALGORITHM				
10								
11								(LSB)
12	Reserved							
13	Reserved							
14	Reserved							
15	Reserved							

See SPC-4 for a description of the PS bit, SPF bit, PAGE CODE field, and PAGE LENGTH field.

A data compression enable (DCE) bit of one specifies data compression is enabled. When this bit is one, data sent to the device server by the application client shall be processed using the selected compression algorithm before being written to the medium. A DCE bit of zero specifies data compression is disabled.

A data compression capable (DCC) bit of one specifies the device supports data compression and is capable of processing data sent to it for transferal to the medium using the selected compression algorithm. A DCC bit of zero specifies the device does not support data compression. This shall be a non-changeable bit.

A data decompression enable (DDE) bit of one specifies data decompression is enabled. A DDE bit of zero specifies data decompression is disabled. Uncompressed data shall be unaffected by the setting of the DDE bit.

The report exception on decompression (RED) field specifies the response to certain boundaries detected in the data on the medium. There are a number of boundaries that may occur on the medium between compressed and

uncompressed data. These boundaries are shown in table 96. Only boundaries shown in table 96 may generate a CHECK CONDITION status.

Table 96 — Possible boundaries and resulting sense keys due to data compression

Prior Data	Current Data	Sense Key[a,b]		
		RED field value		
		zero	one	two
Uncompressed	Compressed (unsupported algorithm)	MEDIUM ERROR	MEDIUM ERROR	MEDIUM ERROR
Uncompressed	Compressed (supported algorithm)	[none]	[none]	RECOVERED ERROR
Compressed (supported algorithm)	Uncompressed	[none]	[none]	NO SENSE
Compressed (supported algorithm)	Compressed (unsupported algorithm)	MEDIUM ERROR	MEDIUM ERROR	MEDIUM ERROR
Compressed (supported algorithm A)	Compressed (supported algorithm B)	[none]	[none]	RECOVERED ERROR
Compressed (unsupported algorithm)	Uncompressed	[none]	NO SENSE	NO SENSE
Compressed (unsupported algorithm)	Compressed (supported algorithm)	[none]	RECOVERED ERROR	RECOVERED ERROR
Compressed (unsupported algorithm A)	Compressed (unsupported algorithm B)	MEDIUM ERROR	MEDIUM ERROR	MEDIUM ERROR
All other combinations		[none]	[none]	[none]

a. [none] specifies no CHECK CONDITION status is returned given the data boundary condition and the current value of the RED field.
b. The appropriate additional sense code is specified following this table in this subclause.

If a CHECK CONDITION status is returned and the current data is compressed, the additional sense code shall be set to either DECOMPRESSION EXCEPTION SHORT ALGORITHM ID OF NN with the additional sense code qualifier set to the algorithm id or DECOMPRESSION EXCEPTION LONG ALGORITHM with no additional sense code qualifier.

If a CHECK CONDITION status is returned and the current data is uncompressed, the additional sense code shall be set to DECOMPRESSION EXCEPTION SHORT ALGORITHM ID OF NN with the additional sense code qualifier set to zero.

A RED field of zero specifies the device shall return a CHECK CONDITION status when data is encountered on the medium during a read operation that the device is unable to decompress. Data boundaries in table 96 marked other than [none] in the column for RED field values of zero shall generate CHECK CONDITION status with the specified sense key when the RED field is zero.

A RED field of one specifies the device shall return a CHECK CONDITION status when data is encountered on the medium during a read operation that requires different handling by the application client than the data most recently encountered during a prior read operation. At each of these boundaries, the data that is sent to the application client is of a fundamentally different nature from that which was previously sent. Data boundaries in table 96

marked other than [none] in the column for RED field values of one shall generate CHECK CONDITION status with the specified sense key when the RED field is one.

A RED field of two specifies the device shall return a CHECK CONDITION status when data is encountered on the medium during a read operation that has been processed using a different algorithm from that data most recently encountered during a prior read operation. Data boundaries in table 96 marked other than [none] in the column for RED field values of two shall generate CHECK CONDITION status with the specified sense key when the RED field is two.

A RED field of three is reserved. If a mode page containing a RED field of three is received, the MODE SELECT command shall be terminated with CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to INVALID FIELD IN PARAMETER LIST.

Upon detection of any of the boundary conditions described in table 96 that results in a CHECK CONDITION status, the additional sense code shall be set to either DECOMPRESSION EXCEPTION SHORT ALGORITHM ID OF NN (if the algorithm identifier is less than or equal to 255) or DECOMPRESSION EXCEPTION LONG ALGORITHM ID. The device shall, in both cases, set the DECOMPRESSION ALGORITHM field to the algorithm identifier of the compression algorithm used to process the encountered data. The logical position shall be on the EOP side of the encountered data, and the INFORMATION field in the sense data shall contain a count of the number of logical blocks contained within the encountered data.

When compressed data is encountered on the medium that the device server is unable to decompress, the device server shall return a CHECK CONDITION status. The sense key shall be set to MEDIUM ERROR and the additional sense code shall be set to CANNOT DECOMPRESS USING DECLARED ALGORITHM. Undecompressed data may be returned to the application client as a single variable length logical block with the ILI bit and INFORMATION fields set accordingly. The logical position is vendor specific following this condition.

NOTE 57 The undecompressed data may contain more than one logical object. As such, the application client should issue a READ POSTION command following this condition to re-establish positioning.

The COMPRESSION ALGORITHM field specifies the currently selected compression algorithm. The default value of the COMPRESSION ALGORITHM field shall specify the default compression algorithm for the device. The field specifies the compression algorithm the device shall use to process data sent to it by the application client when the DCE bit is set to one. If the application client selects an algorithm that the device does not support, then the device shall return a CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN PARAMETER LIST. Algorithm identifiers are shown in table 97. The SELECT DATA COMPRESSION ALGORITHM field in the Device Configuration mode page shall be ignored if a Data Compression mode page with the DCE bit set to one is also received by the device in the same MODE SELECT command.

For the MODE SELECT command, the DECOMPRESSION ALGORITHM field specifies the decompression algorithm selected by the application client for use in subsequent decompression of data encountered on the medium. For devices capable of the automatic recognition of the compression algorithm used to process data encountered on the medium, the decompression algorithm selected by the application client may be ignored, or overridden by the logical unit for a subsequent read operation if the selected value does not match the compression algorithm that was used to process the data encountered on the medium.

For the MODE SENSE command, the DECOMPRESSION ALGORITHM field reflects the algorithm selected by the application client. For some devices, the DECOMPRESSION ALGORITHM value returned in response to a MODE SENSE command may change dynamically to match the compression algorithm, detected by the device, that was used to process the data most recently encountered on the medium, during a read operation. A value of zero specifies the

data encountered on the medium during the most recent read operation was uncompressed. Compression algorithm identifiers are shown in table 97.

Table 97 — Compression algorithm identifiers

Algorithm Identifier	Description
00h	No algorithm selected (i.e., identifies uncompressed data).
01h	Set with MODE SELECT to select the default algorithm. MODE SENSE shall return the actual compression algorithm that was selected.
02h	Reserved.
03h	IBM ALDC[a] data compression algorithm with 512 byte buffer.
04h	IBM ALDC[a] data compression algorithm with 1024 byte buffer.
05h	IBM ALDC[a] data compression algorithm with 2048 byte buffer.
06h - 0Fh	Reserved.
10h	IBM IDRC[b] data compaction algorithm.
11h - 1Fh	Reserved.
20h	DCLZ[c] data compression algorithm.
21h - FEh	Reserved.
FFh	Unregistered algorithm.
100h - FFFFFFFFh	Reserved.

a. Adaptive Lossless Data Compression (see ISO/IEC 15200:1996).
b. Improved Data Recording Capability
c. Data Compression according to Lempel and Ziv (see ISO/IEC 11558:1992).

8.3.3 Device Configuration mode page

The Device Configuration mode page (see table 98) is used to specify the appropriate sequential-access device configuration.

Table 98 — Device Configuration mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF(0)	PAGE CODE (10h)					
1	PAGE LENGTH (0Eh)							
2	Rsvd	Obsolete	CAF	ACTIVE FORMAT				
3	ACTIVE PARTITION							
4	WRITE OBJECT BUFFER FULL RATIO							
5	READ OBJECT BUFFER EMPTY RATIO							
6	(MSB)	WRITE DELAY TIME						
7								(LSB)
8	OBR	LOIS	Obsolete	AVC	SOCF		ROBO	REW
9	Obsolete							
10	EOD DEFINED			EEG	SEW	SWP	BAML	BAM
11	(MSB)	OBJECT BUFFER SIZE AT EARLY WARNING						
12								
13								(LSB)
14	SELECT DATA COMPRESSION ALGORITHM							
15	WTRE		OIR	REWIND ON RESET		ASOCWP	PERSWP	PRMWP

See SPC-4 for a description of the PS bit, SPF bit, PAGE CODE field, and PAGE LENGTH field.

NOTE 58 The change active partition (CAP) bit (byte 2, bit 6 in the Device Configuration mode page) has been obsoleted. To change active partitions refer to the LOCATE command.

A change active format (CAF) bit of one specifies the active format is to be changed to the value specified in the ACTIVE FORMAT field. A CAF bit of zero specifies no active format change is specified. For some devices, the format may only be changed when the logical unit is at beginning-of-partition.

The ACTIVE FORMAT field specifies the recording format that is in use for the selected density code when reading or writing data on a logical unit. The value of the ACTIVE FORMAT field is vendor specific.

The ACTIVE PARTITION field specifies the current logical partition number in use on the medium. This shall be a non-changeable field.

The WRITE OBJECT BUFFER FULL RATIO field, on WRITE commands, specifies to the device server how full the object buffer shall be before writing data to the medium. A value of zero specifies the value is not specified.

The READ OBJECT BUFFER EMPTY RATIO field, on READ commands, specifies to the device server how empty the object buffer shall be before retrieving additional data from the medium. A value of zero specifies the value is not specified.

The WRITE DELAY TIME field specifies the maximum time, in 100 ms increments, that the device server should wait before any buffered data that is to be written, is forced to the medium after the last buffered WRITE command that did not cause the object buffer to exceed the write object buffer full ratio. A value of zero specifies the device server shall never force buffered data to the medium under these conditions.

An object buffer recovery (OBR) bit of one specifies the logical unit supports object buffer recovery using the RECOVER BUFFERED DATA command. An OBR bit of zero specifies the logical unit does not support object buffer recovery. Most device servers consider this bit to be not changeable.

A logical object identifiers supported (LOIS) bit of zero specifies logical object identifiers are not supported in the format written on the medium. A LOIS bit of one specifies the format on the medium has recorded information about the logical object identifiers relative to a partition. Most device servers consider this bit to be not changeable.

The automatic velocity control (AVC) bit of one, specifies the device shall select the speed (if the device supports more than one speed) based on the data transfer rate that should optimize streaming activity and minimize medium repositioning. An AVC bit of zero specifies the speed chosen shall be defined by the SPEED field in the mode parameter header.

A stop on consecutive filemarks (SOCF) field of 00b specifies the device server shall pre-read data from the medium to the limits of the object buffer capacity without regard for filemarks. Values 01b, 10b, and 11b specify that the device server shall terminate the pre-read operation if one, two, or three consecutive filemarks are detected, respectively.

A recover object buffer order (ROBO) bit of one specifies logical blocks shall be returned from the object buffer of the logical unit on a RECOVER BUFFERED DATA command in LIFO order (last-in-first-out) from that they were written to the object buffer. A RBO bit of zero specifies logical blocks shall be returned in FIFO (first-in-first-out) order.

A report early-warning (REW) bit of zero specifies the device server shall not report the early-warning condition for read operations and it shall report early-warning at or before any medium-defined early-warning position during write operations.

A REW bit of one specifies the device server shall return CHECK CONDITION status. The additional sense code shall be set to END-OF-PARTITION/MEDIUM DETECTED, and the EOM bit set to one in the sense data when the early-warning position is encountered during read and write operations. If the REW bit is one and the SEW bit is zero, the device server shall return CHECK CONDITION status with the sense key set to VOLUME OVERFLOW when early-warning is encountered during write operations.

NOTE 59 A REW bit of one is intended for compatibility with application clients using old tape formats that require an early-warning indication during read operations. Other application clients should set this bit to zero to avoid potential data loss when interchanging tapes between devices.

The EOD DEFINED field specifies the format type that the logical unit shall use to detect and generate the EOD area. The values for EOD DEFINED are specified in table 99.

Table 99 — EOD DEFINED values

Code	Description
000b	Logical unit's default EOD definition
001b	Format-defined erased area of medium
010b	As specified in the SOCF field
011b	EOD recognition and generation is not supported
100b - 111b	Reserved

An enable EOD generation (EEG) bit set to one specifies the logical unit shall generate the appropriate EOD area, as determined by the EOD field. A value of zero specifies EOD generation is disabled.

NOTE 60 Some logical units may not generate EOD at the completion of any write-type operation.

A synchronize at early-warning (SEW) bit set to one specifies the logical unit shall cause any buffered logical objects to be transferred to the medium prior to returning status when positioned between early-warning and EOP. A SEW bit of zero specifies the logical unit may retain unwritten buffered logical objects in the object buffer when positioned between early-warning and EOP (see 5.6, 5.7, 6.8, and 6.9).

A software write protection (SWP) bit set to one specifies the device server shall perform a synchronize operation then enter the write-protected state (see 4.2.13 and 4.2.13.3). When the SWP bit is set to one, all commands requiring eventual writes to the medium shall return CHECK CONDITION status. The sense key shall be set to DATA PROTECT and the additional sense code should be set to LOGICAL UNIT SOFTWARE WRITE PROTECTED (see 4.2.13.2). A SWP bit set to zero specifies the device server may inhibit writing to the medium, dependent on other write inhibits.

A block address mode lock (BAML) bit of zero specifies the selection of the block address mode shall be determined based on the first block address mode unique command that is received after a successful load operation or a successful completion of a command that positions the medium to BOP. A BAML bit of one specifies the selection of the block address mode shall be determined based on the setting of the BAM bit. See 4.2.16 for a description of block address mode selection.

The block address mode (BAM) bit is valid only if the BAML bit is set to one. If the BAML bit is set to zero, the BAM bit shall be ignored. If the BAML bit is set to one and the BAM bit is set to zero, the logical unit shall operate using implicit address mode. If the BAML bit is set to one and the BAM bit is set to one, the logical unit shall operate using explicit address mode. See 4.2.16 for a description of block address mode selection.

The OBJECT BUFFER SIZE AT EARLY WARNING field specifies the value, in bytes, that the logical unit shall reduce its logical object buffer size to when writing in a position between its early-warning and end-of-partition. A value of zero specifies the implementation of this function is vendor specific.

NOTE 61 The intent is to prevent the loss of data by limiting the size of the object buffer when near the end-of-partition.

The SELECT DATA COMPRESSION ALGORITHM field set to 00h specifies the logical unit shall not use a compression algorithm on any data sent to it prior to writing the data to the medium. A value of 01h specifies the data to be written shall be compressed using the logical unit's default compression algorithm. Values 02h through 7Fh are reserved. Values 80h through FFh are vendor specific. The SELECT DATA COMPRESSION ALGORITHM field shall be ignored if a Data Compression mode page with the DCE bit set to one is received by the device in the same MODE SELECT command.

NOTE 62 New implementations use the Data Compression mode page (see 8.3.2) for specifying data compression behavior.

The WORM Tamper Read Enable (WTRE) field specifies how the device server responds to detection of comprimised integrity of a WORM medium when processing a locate, read, read reverse, space, or verify operation. The

WTRE field shall have no effect on the processing of a locate, read, read reverse, space, or verify operation when the device contains a non-WORM medium. The values for the WTRE field are specified in table 100.

Table 100 — WTRE field definition

Code	Description
00b	The device server shall respond in a vendor-specific manner.
01b	Detection of comprimised integrity on a WORM medium shall not affect processing of a task.
10b	The device server shall return CHECK CONDITION status. The sense key shall be set to MEDIUM ERROR and the additional sense code shall be set to WORM MEDIUM - INTEGRITY CHECK. The position of the medium may have changed,
11b	Reserved. The device server shall return CHECK CONDITION status for a MODE SELECT command with the WTRE field set to 11b. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN PARAMETER LIST.

NOTE 63 An application client should set the WTRE field to 01b only for the recovery of data from a WORM medium where the integrity of the stored data has been comprimised.

If the only if reserved (OIR) bit is set to one, the device server shall process a command only if a reservation (see SPC-2) or persistent reservation (see SPC-4) exists that allows access via the I_T nexus from which the command was received. If the OIR bit is set to one and a command is received from an I_T nexus for which no reservation exists, the device server shall not process the command. If the OIR bit is set to one and a command is received from an I_T nexus for a logical unit upon which no reservation or persistent reservation exists, the device server shall terminate the command with CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to NOT RESERVED. Commands that shall not be effected by the OIR bit set to one are defined as Allowed in the presence of persistent reservations in table 14 or SPC-4, or are defined in SPC-2 as Allowed in the presence of reservations. Commands that shall be effected by the OIR bit set to one are defined as Conflict in the presence of persistent reservations in table 14 or SPC-4, except for the RESERVE, RELEASE, PERSISTENT RESERVATION IN and PERSISTENT RESERVATION OUT commands, or are defined in SPC-2 as Conflict in the presence of reservations. An OIR bit set to zero specifies the device server shall process commands as specified in table 14 or SPC-4.

The REWIND ON RESET field is specified in table 101. The REWIND ON RESET field, if implemented, shall be persistent across logical unit resets.

Table 101 — REWIND ON RESET field definition

Code	Description
00b	Vendor specific
01b	The logical unit shall position to the beginning of the default data partition (BOP 0) on logical unit reset.
10b	The logical unit shall maintain its position on logical unit reset.
11b	Reserved

An associated write protection (ASOCWP) bit set to one specifies the logical unit shall inhibit all writing to the medium after performing a synchronize operation (see 4.2.13 and 4.2.13.4). When the ASOCWP bit is set to one, the currently mounted volume is logically write protected until the volume is demounted (see 4.2.13 and 4.2.13.4). When the ASOCWP bit is set to one, all commands requiring eventual writes to the medium shall return CHECK

CONDITION status. The sense key shall be set to DATA PROTECT and the additional sense code should be set to ASSOCIATED WRITE PROTECT (see 4.2.13.2). An ASOCWP bit set to zero specifies the currently mounted volume is not write protected by the associated write protection. The ASOCWP bit shall be set to zero by the device server when the volume is demounted. This change of state shall not cause a unit attention condition. If the application client sets the ASOCWP bit to one while no volume is mounted, the device server shall terminate the MODE SELECT command with CHECK CONDITION status. The sense key shall be set to NOT READY and the additional sense code shall be set to MEDIUM NOT PRESENT. If the Device Configuration mode page is savable, the ASOCWP bit shall be saved as zero, regardless of the current setting.

A persistent write protection (PERSWP) bit set to one specifies the currently mounted volume is logically write protected (see 4.2.13 and 4.2.13.5). When the PERSWP bit is set to one, all commands requiring eventual writes to the medium shall return CHECK CONDITION status. The sense key shall be set to DATA PROTECT and the additional sense code should be set to PERSISTENT WRITE PROTECT (see 4.2.13.2). A PERSWP bit set to zero specifies the currently mounted volume is not write protected by the persistent write protection. The PERSWP bit shall be set to zero by the device server when the volume is demounted or when a volume is mounted with persistent write protection disabled. The PERSWP bit shall be set to one by the device server when a volume is mounted with persistent write protection enabled. These changes of state shall not cause a unit attention condition. If the application client sets the PERSWP bit to one while no volume is mounted, the device server shall terminate the MODE SELECT command with CHECK CONDITION status. The sense key shall be set to NOT READY and the additional sense code shall be set to MEDIUM NOT PRESENT. If the application client sets the PERSWP bit to one when the logical position is not at BOP 0, the device server shall return CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to POSITION PAST BEGINNING OF MEDIUM. If the Device Configuration mode page is savable, the PERSWP bit shall be saved as zero, regardless of the current setting.

A permanent write protection (PRMWP) bit set to one specifies the currently mounted volume is logically write protected (see 4.2.13 and 4.2.13.6). When the PRMWP bit is set to one, all commands requiring eventual writes to the medium shall return CHECK CONDITION status. The sense key shall be set to DATA PROTECT and the additional sense code should be set to PERMANENT WRITE PROTECT (see 4.2.13.2). A PRMWP bit set to zero specifies the currently mounted volume is not write protected by the permanent write protection. The PRMWP bit shall be set to zero by the device server when the volume is demounted or when a volume is mounted with permanent write protection disabled. The PRMWP bit shall be set to one by the device server when a volume is mounted with permanent write protection enabled. These changes of state shall not cause a unit attention condition. If the application client sets the PRMWP bit to one while no volume is mounted, the device server shall terminate the MODE SELECT command with CHECK CONDITION status. The sense key shall be set to NOT READY and the additional sense code shall be set to MEDIUM NOT PRESENT. If the application client sets the PRMWP bit to one when the logical position is not at BOP 0, the device server shall return CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to POSITION PAST BEGINNING OF MEDIUM. If the application client attempts to change the PRMWP bit from one to zero, the device server shall terminate the MODE SELECT command with CHECK CONDITION status. The sense key shall be set to DATA PROTECT and the additional sense code shall be set to PERMANENT WRITE PROTECT. If the Device Configuration mode page is savable, the PRMWP bit shall be saved as zero, regardless of the current setting.

8.3.4 Medium Partition mode page

The Medium Partition mode page (see table 102) is used to specify the group of medium partitions. Fields in the Medium Partition mode page indicating the current state of the partitions for the medium shall be changed by the device server to the current medium state on a not ready to ready transition when the medium state changes from demounted to mounted. The physical placement and order of medium partitions are not specified by this standard.

NOTE 64 Since defining partitions may require reformatting the medium for some implementations, an implicit write to the medium may occur as a result of a MODE SELECT command that supplies these parameters.

Table 102 — Medium Partition mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF(0)	PAGE CODE (11h)					
1	PAGE LENGTH							
2	MAXIMUM ADDITIONAL PARTITIONS							
3	ADDITIONAL PARTITIONS DEFINED							
4	FDP	SDP	IDP	PSUM		POFM	CLEAR	ADDP
5	MEDIUM FORMAT RECOGNITION							
6	Reserved				PARTITION UNITS			
7	Reserved							
Partition size descriptor(s)								
8	(MSB)	PARTITION SIZE						
9								(LSB)
n-1	(MSB)	PARTITION SIZE						
n								(LSB)

See SPC-4 for a description of the PS bit, SPF bit, PAGE CODE field, and PAGE LENGTH field.

The MAXIMUM ADDITIONAL PARTITIONS field is a logical unit-defined value indicating the maximum number of additional partitions supported by the logical unit. A value of zero returned by the MODE SENSE command specifies no additional partitions are present or allowed.

The ADDITIONAL PARTITIONS DEFINED field specifies the number of additional partitions to be defined for a volume when the SDP or IDP bit is set to one. The maximum value allowed is the value returned in the MAXIMUM ADDITIONAL PARTITIONS field. The ADDITIONAL PARTITIONS DEFINED value returned by the MODE SENSE command shall report one less than the number of partitions on the media when the logical unit is ready. If the unit is not ready, the ADDITIONAL PARTITIONS DEFINED field is undefined.

A fixed data partitions (FDP) bit of one specifies the logical unit shall partition the medium based on its fixed definition of partitions if the POFM bit is set to zero. Setting this bit to one when POFM is set to zero may only be valid at beginning-of-partition and is mutually exclusive with the SDP and IDP bits. The partition size descriptors are ignored by the MODE SELECT command when the FDP bit is set to one. The logical unit may assign any number of partitions from 1 to (MAXIMUM ADDITIONAL PARTITIONS + 1).

NOTE 65 It is recommended that the partition size descriptors be present in MODE SENSE data regardless of the settings of the FDP, SDP or IDP fields to give an estimate of the size of each partition.

A select data partitions (SDP) bit of one specifies the logical unit shall partition the medium into the number of partitions as specified by the ADDITIONAL PARTITIONS DEFINED field (*n*) using partition sizes defined by the device if the POFM bit is set to zero. The logical unit shall partition the medium into *n*+1 partitions numbered 0 through *n*. Setting this bit to one when POFM is set to zero may only be valid at beginning-of-partition and it is mutually exclusive with

the FDP and IDP fields. The partition size descriptors are ignored by the MODE SELECT command when the SDP bit is set to one.

An initiator-defined partitions (IDP) bit of one specifies the logical unit shall partition the medium as defined by the ADDITIONAL PARTITIONS DEFINED field and the partition size descriptors if the POFM bit is set to zero. Setting this bit to one when POFM is set to zero may only be valid at beginning-of-partition and is mutually exclusive with the FDP and SDP fields. The number of non-zero partition size descriptors received in the Medium Partition mode page shall be one more than the ADDITIONAL PARTITIONS DEFINED value. The size of partition 0 shall be non-zero.

A logical unit is not required to retain the method used to partition the medium. The device server shall set only one of the IDP, FDP or SDP fields in the MODE SENSE data. If a volume was previously partitioned through a MODE SELECT command with FDP or SDP set to one, a device server may set IDP to one in subsequent MODE SENSE data since the volume has been initiator partitioned. However, in a MODE SELECT command, the application client cannot use IDP set to one in place of FDP or SDP set to one.

NOTE 66 Since defining partitions may require reformatting the medium for some implementations, an implicit write to the medium may occur as a result of a MODE SELECT command that has any of the fields FDP, SDP, or IDP set to one and has a value of zero in the POFM field."

The partition size unit of measure (PSUM) field defines the units of the partition size descriptors. A logical unit is not required to retain the partition size unit of measure used to partition the medium. The PSUM field is defined in table 103.

Table 103 — PSUM values

Code	Description	Support
00b	bytes (unit of one)	O
01b	kilobytes (10^3 bytes)	O
10b	megabytes (10^6 bytes)	O
11b	$10^{(\text{PARTITION UNITS})}$ bytes	O

The PARTITION UNITS field defines the size of the partition size descriptors when the PSUM field is set to 11b. A value of *n* in the PARTITION UNITS field shall define the units of the partition size descriptors as 10^n bytes. If the PARTITION UNITS field is supported, all possible values shall be supported. A logical unit is not required to retain the partition units used to partition the medium. If PSUM is not equal to 11b, the PARTITION UNITS field is undefined. Some values of the PARTITION UNITS field may result in no legal non-zero partition size descriptors.

A partition on format (POFM) bit of one specifies the MODE SELECT command shall not cause changes to the partition sizes or user data, either recorded or buffered. If POFM is set to one, actual media partitioning occurs when the device server receives a subsequent FORMAT MEDIUM command (see 7.1). When the FORMAT MEDIUM command partitions the media, it shall do so based on the contents of the mode data for the Medium Partition mode page. If POFM is set to one, field values specified by a MODE SELECT command for the Medium Partition mode page shall not be changed by the device server before the media is unloaded or until a logical unit reset. Some field checking may be performed by the MODE SELECT command. However, there is no guarantee that any subsequent partitioning during a FORMAT MEDIUM command will complete with no errors.

A POFM bit of zero specifies the MODE SELECT command shall alter the partition information for the medium if any of the SDP, FDP, or IDP bits are set to one.

A CLEAR bit of zero and an ADDP bit of zero specifies the logical unit may logically erase any or all partitions when one of the IDP, FDP, or SDP fields are set to one by a MODE SELECT command.

A CLEAR bit of one and an ADDP bit of zero specifies the logical unit shall logically erase every partition if one of the IDP, FDP, or SDP fields is set to one. No formatting of the medium is implied.

An ADDP bit of one and a CLEAR bit of zero specifies the logical unit shall not logically erase any existing partitions, even if the size of the partition is changed. If the MODE SELECT command partition size descriptor and the current partition size differ, the logical unit shall truncate or extend the partition, whichever is appropriate. If the MODE SELECT command partition size is zero and the current partition size is non-zero, the partition shall be logically removed from the medium, resulting in the loss of all data in that partition. If the MODE SELECT command partition size is equivalent to the current partition size, no change in the partition size shall result. If the logical unit is unable to perform the operation or if such an operation would cause loss of valid data in any partition that exists both before and after the MODE SELECT or FORMAT MEDIUM command, the device server shall return CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST with the addition sense code set to PARAMETER VALUE INVALID. If the ADDP bit is set to one and either ADDP is not supported or the FDP field is set to one the device server shall return CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN PARAMETER LIST. If both the ADDP and SDP fields are set to one, the logical unit shall add or remove partitions such that the resulting partition count on the medium is equal to the ADDITIONAL PARTITIONS DEFINED value plus one.

If both the ADDP and CLEAR fields are set to one, the logical unit shall logically erase all partitions that differ in size from the corresponding partition size descriptor in the MODE SELECT data. Partitions with the same size as the MODE SELECT data size shall retain all existing data. If the logical unit is incapable of supporting the changes requested without loss of data, the device server shall return CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to PARAMETER VALUE INVALID. If setting both ADDP and CLEAR to one is not supported, the sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN PARAMETER LIST.

A MODE SELECT command partition size descriptor has the equivalent (same) size as the current partition size if:

a) the mode select PARTITION SIZE, PSUM, and PARTITION UNITS fields are exactly the same as those returned by MODE SENSE command;
b) the mode select PARTITION SIZE field value is within plus or minus one of the current size when the current size is converted to the units of the mode select PSUM or PARTITION UNITS field; or
c) the mode select PARTITION SIZE is FFFFh and the current size would return FFFFh if expressed in the units of the mode select PSUM or PARTITION UNITS field.

The MEDIUM FORMAT RECOGNITION field specifies the logical unit's capability to automatically identify the medium format and partition information when reading a volume. The value in this field may be different following a medium change. The MEDIUM FORMAT RECOGNITION field values are shown in table 104.

Table 104 — Medium format recognition values

Code	Description
00h	Logical unit is incapable of format or partition recognition.
01h	Logical unit is capable of format recognition.
02h	Logical unit is capable of partition recognition.
03h	Logical unit is capable of format and partition recognition.
04h - FFh	Reserved

NOTE 67 If a logical unit specifies it is not capable of medium format recognition, the application client should supply all necessary parameters for the device to identify the specific format.

PARTITION SIZE fields within the partition size descriptor list define the approximate size of the respective partitions in the units specified in the PSUM and PARTITION UNITS fields. Partitions are numbered by their relative position in the partition size descriptor list, starting at 0. Only partition numbers in the range of 0 to *n* where *n* is less than or equal to 63 may have size descriptors in this mode page. Partition *n*, if present, shall be described by the partition size descriptor at mode page offsets 8+(2*n*) and 9+(2*n*). Partition 0 shall be the default partition. Partition size descriptor 0, shall contain the size of the default partition. The size of partition 0 shall be greater than 0. Up to 64 partitions may be defined using this mode page. Partitions not assigned shall have a partition size descriptor of 0. The logical unit may support more partitions than partition size descriptors. A logical unit may support more partition size descriptors than supported by the medium. All partition size descriptors representing a partition number greater than the maximum additional partition count shall be 0. The partition size descriptors are undefined if the logical unit is not ready. A MODE SELECT command partition size descriptor of FFFFh requests that the logical unit allocate all remaining partition space to that partition. A MODE SENSE command shall return a partition size descriptor of FFFFh if the partition size, in units of PSUM or PARTITION UNITS, is greater than or equal to FFFFh. If insufficient space exists on the medium for the requested partition sizes or if multiple partition size descriptors are set to FFFFh, the device server shall return CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN PARAMETER LIST. A device server may round, as described by the MODE SELECT command in SPC-4, any partition size to the nearest valid partition size.

NOTE 68 It is recommended, but not required, that the number of partition size descriptors available through the Medium Partition mode page equal at least the number of maximum addition partitions + 1. This provides a mechanism for the device server to disclose the current partition sizes.

8.3.5 Read-Write Error Recovery mode page

The Read-Write Error Recovery mode page (see table 105) specifies the error recovery and reporting parameters that the device server shall use when transferring data between the device and the medium. These parameters do not affect protocol-level recovery procedures or positioning error recovery procedures.

Table 105 — Read-Write Error Recovery mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF(0)	PAGE CODE (01h)					
1	PAGE LENGTH (0Ah)							
2	Reserved		TB	Rsvd	EER	PER	DTE	DCR
3	READ RETRY COUNT							
4	Reserved							
5	Reserved							
6	Reserved							
7	Reserved							
8	WRITE RETRY COUNT							
9	Reserved							
10	Reserved							
11	Reserved							

NOTE 69 The parameters in the Read-Write Error Recovery mode page also apply to verify operations.

See SPC-4 for a description of the PS bit, SPF bit, PAGE CODE field, and PAGE LENGTH field.

A transfer block (TB) bit of one specifies the device server shall use its best effort to transfer a logical block that cannot be read successfully within the specified read recovery limits to the application client before CHECK CONDITION status is returned. A TB bit of zero specifies an unrecoverable logical block shall not be transferred to the application client. Logical blocks that are recoverable within the recovery limits are always transferred, regardless of the value of the TB bit.

An enable early recovery (EER) bit of one specifies the logical unit shall use the most expedient error recovery algorithm (e.g., attempt error correction prior to retries). An EER bit of zero specifies the logical unit shall use the most deliberate error recovery algorithm, within the limits established by the other error recovery parameters (e.g., attempt to recover the logical block error-free prior to using error correction).

A post error (PER) bit of one specifies the device server shall return CHECK CONDITION status to report recovered errors. A PER bit of zero specifies the device server shall not report errors recovered within the limits established by the error recovery parameters. If this bit is zero, the DTE bit shall also be set to zero.

A disable transfer on error (DTE) bit of one specifies the device server shall terminate the data transfer after a recovered read or write error occurs. All data from the recovered logical block shall be transferred prior to terminating the read or write operation. A DTE bit of zero specifies the device server shall not terminate the transfer for errors recovered within the limits established by the read-write error recovery parameters.

A disable correction (DCR) bit of one specifies the logical unit shall not use error correction codes during error recovery. A DCR bit of zero allows the use of error correction codes for error recovery.

The READ RETRY COUNT field specifies the number of times that the logical unit should attempt its recovery algorithm during a read operation before an unrecoverable error is reported. A READ RETRY COUNT of zero specifies the logical unit shall not use its recovery algorithm during read operations.

The WRITE RETRY COUNT field specifies the number of times that the logical unit should attempt its recovery algorithm during a write operation before an unrecoverable error is reported. A WRITE RETRY COUNT of zero specifies the logical unit shall not use its recovery algorithm during write operations.

8.3.6 Informational Exceptions Control mode page

In addition to support for all device types (see SPC-4), the Informational Exceptions Control mode page (see table 106) specifies the parameters for the control of TapeAlert specific informational exception conditions for a sequential-access device.

Table 106 — Informational Exceptions Control mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF (0)	PAGE CODE (1Ch)					
1	PAGE LENGTH (0Ah)							
2	PERF	Rsvd	EBF	EWASC	DEXCPT	TEST	Rsvd	LOGERR
3	Reserved				MRIE			
4	(MSB)	INTERVAL TIMER						
5								
6								
7								(LSB)
8	(MSB)	REPORT COUNT/TEST FLAG NUMBER						
9								
10								
11								(LSB)

See SPC-4 for a description of the PS bit, SPF bit, PAGE CODE field, PAGE LENGTH field, PERF bit, EBF bit, EWASC bit, LOGERR bit, and INTERVAL TIMER field.

SPC-4 defines the effect of setting the TEST bit to one if the REPORT COUNT/TEST FLAG NUMBER field is set to zero. In response to a MODE SENSE command reporting parameters from the Informational Exceptions Control mode page, the device server shall set the value of the TEST bit to zero. The device server shall not alter the value of any TapeAlert flags in response to an application client setting the TEST bit to one and the REPORT COUNT/TEST FLAG NUMBER field to zero.

Table 71 defines the effect of setting the TEST bit to one if the REPORT COUNT/TEST FLAG NUMBER field is set to a non-zero value. In response to a MODE SENSE command reporting parameters from the Informational Exceptions Control mode page, the device server shall set the value of the TEST bit to zero. If both the TEST bit and the DEXCEPT bit are set to one, the device server shall terminate the MODE SELECT command with CHECK CONDITION

status, set the sense key set to ILLEGAL REQUEST, and set the additional sense code set to INVALID FIELD IN PARAMETER LIST.

Table 107 — TEST bit and TEST FLAG NUMBER field definition

TEST FLAG NUMBER	Description
1 to 64	Activate the TapeAlert flag specified by the TEST FLAG NUMBER field. Report the informational exception condition for the TapeAlert flag with an additional sense code of FAILURE PREDICTION THRESHOLD EXCEEDED (FALSE) and based on the DEXCPT, MRIE, INTERVAL TIMER, and REPORT COUNT values.
-1 to -64	Deactivate the TapeAlert flag specified by the absolute value of the TEST FLAG NUMBER field. Deactivating the flag in this way is equivalent to performing the specified corrective action for that flag.
32767	Activate all supported TapeAlert flags. Report the informational exception condition for the TapeAlert flag with an additional sense code of FAILURE PREDICTION THRESHOLD EXCEEDED (FALSE) and based on the DEXCPT, MRIE, INTERVAL TIMER, and REPORT COUNT values.
all others	Return CHECK CONDITION status. Set the sense key to ILLEGAL REQUEST and the additional sense code to INVALID FIELD IN PARAMETER LIST.

SPC-4 specifies the effect of setting the TEST bit to zero.

See SPC-4 for a description of the DEXCPT bit. A device server shall not report non-TapeAlert informational exceptions with an additional sense code of FAILURE PREDICTION THRESHOLD EXCEEDED if the DEXCPT bit is set to zero and the taser bit in the Device Configuration Extension mode page is set to zero (see clause 8.3.8).

See SPC-4 for a description of the MRIE field. For MRIE modes 02h to 06h, an additional sense code of FAILURE PREDICTION THRESHOLD EXCEEDED specifies a TapeAlert event has occurred on the device. Detailed information about the event is stored in the TapeAlert log page. If multiple TapeAlert flags are active simultaneously, the device server shall report a single informational exception condition.

NOTE 70 The value of the MRIE field does not affect parameters in the TapeAlert log page or the TapeAlert Response log page.

The REPORT COUNT/TEST FLAG NUMBER field has a dual purpose:

a) SPC-4 specifies the operation of the REPORT COUNT/TEST FLAG NUMBER field if the TEST bit is set to zero. When reporting an informational exception condition associated with TapeAlert flags, upon activation of a TapeAlert flag the device server shall report to the application client the informational exception condition the number of times indicated by the value of the REPORT COUNT/TEST FLAG NUMBER field; and
b) if the TEST bit is set to one, the value of the REPORT COUNT/TEST FLAG NUMBER field represents the TEST FLAG NUMBER. In response to a MODE SENSE command reporting parameters from the Informational Exceptions Control mode page, the device server shall set the value of the REPORT COUNT/TEST FLAG NUMBER field to zero. Table 107 specifies valid values for the TEST FLAG NUMBER field. Negative numbers shall be represented using the 2's complement notation and shall be sign extended to 4 bytes.

8.3.7 Medium Configuration mode page

The Medium Configuration mode page (see table 108) specifies any special considerations the device server shall use when processing commands that access the medium.

Table 108 — Medium Configuration mode page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	SPF(0)	PAGE CODE (1Dh)					
1	PAGE LENGTH (1Eh)							
2	Reserved							WORMM
3	Reserved							
4	WORM MODE LABEL RESTRICTIONS							
5	WORM MODE FILEMARK RESTRICTIONS							
6-31	Reserved							

See SPC-4 for a description of the PS bit, SPF bit, PAGE CODE field, and PAGE LENGTH field.

The WORM mode (WORMM) bit shall be set to one when the device server is operating in WORM mode (see 4.2.20.3). The WORMM bit shall be set to zero when the device server is not operating in WORM mode. If a MODE SELECT command is processed that attempts to change to setting of the WORMM bit, the device server shall return CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID FIELD IN PARAMETER LIST.

The WORM MODE LABEL RESTRICTIONS field specifies the restrictions against overwriting format labels when operating in WORM mode (see table 109). A series of filemarks with no interleaved logical blocks immediately preceding EOD is treated as a filemark sequence and is controlled by the WORM MODE FILEMARKS RESTRICTIONS field.

Table 109 — WORM MODE LABEL RESTRICTIONS field values

Value	Description
00h	The device server shall not allow any logical blocks to be overwritten.
01h	The device server shall not allow some types of format labels to be overwritten.
02h	The device server shall allow all format labels to be overwritten.
03h - FFh	Reserved

The WORM MODE FILEMARKS RESTRICTIONS field specifies the restrictions against overwriting a series of filemarks immediately preceding EOD when operating in WORM mode (see table 110). The WORM MODE FILEMARKS RESTRIC-

TIONS field controls only the overwriting of a series of filemarks with no interleaved logical blocks immediately preceding EOD.

Table 110 — WORM MODE FILEMARKS RESTRICTIONS field values

Value	Description
00h - 01h	Reserved
02h	The device server shall allow any number of filemarks immediately preceding EOD to be overwritten except the filemark closest to BOP.
03h	The device server shall allow any number of filemarks immediately preceding EOD to be overwritten.
04h - FFh	Reserved

8.3.8 Device Configuration Extension mode page

The Device Configuration Extension mode page (see table 111), a subpage of the Device Configuration mode page (see 8.3.3), provides control of the SCSI features specific to sequential-access devices. If a device server supports the Device Configuration Extension mode page, the device server shall provide access to the mode page using the shared mode page policy (see SPC-4).

Table 111 — Device Configuration Extension mode page

Bit Byte	7	6	5	4	3	2	1	0
0	PS	SPF(1b)	PAGE CODE (10h)					
1	SUBPAGE CODE (01h)							
2	(MSB)	PAGE LENGTH (1Ch)						
3								(LSB)
4	Reserved				TARPF	TASER	TARPC	TAPLSD
5	Reserved				SHORT ERASE MODE			
6	(MSB)	PEWS						
7								(LSB)
8	Reserved							VCELBRE
9-31	Reserved							

See SPC-4 for a description of the PS bit, SPF bit, PAGE CODE field, and PAGE LENGTH field.

A TapeAlert Prevent LOG SENSE Deactivation (TAPLSD) bit set to one specifies that the device server shall not alter the value of implemented TapeAlert FLAG parameters (see 8.2.3) due to processing of a LOG SENSE command with the PAGE CODE field set to 2Eh. A TAPLSD bit set to zero specifies that, as part of the processing of a LOG SENSE command with the PAGE CODE field set to 2Eh, the device server shall deactivate every supported TapeAlert flag.

A TapeAlert Respect Page Control (TARPC) bit set to one specifies that the device server shall select the type of parameter values for the TapeAlert log page (see 8.2.3) using the value of the PC field in a LOG SELECT or LOG SENSE command (see SPC-4). A TARPC bit set to zero specifies that the device server shall select cumulative parameter values for the TapeAlert log page regardless of the value of the PC field in a LOG SELECT or LOG SENSE command.

NOTE 71 An application client using the TapeAlert threshold usage model (see 4.2.17.2.4) should set the TARPC bit to one.

A TapeAlert Select Exception Reporting (TASER) bit set to one specifies that:

a) activation of a TapeAlert flag shall not result in an informational exception condition (see 8.3.6); and
b) activation or deactivation of a TapeAlert flag may result in the generation of a unit attention condition with the additional sense code set to THRESHOLD CONDITION MET depending on the value of the ETC bit in each parameter in the TapeAlert log page.

A TASER bit set to zero specifies that:

a) activation of a TapeAlert flag shall result in an informational exception condition (see 8.3.6);
b) activation or deactivation of a TapeAlert flag shall not result in the generation of a unit attention condition with the additional sense code set to THRESHOLD CONDITION MET; and
c) the device server shall set the ETC bit in each parameter of the TapeAlert log page to zero.

The device server may provide independent sets of TapeAlert log parameters for each I_T nexus (see SPC-4). If the device server does not support independent sets of TapeAlert log parameters and the processing of a MODE SELECT command with the TASER bit set to zero results in a change in the value of the ETC bit in a TapeAlert log parameter, then the device server shall establish a unit attention condition (see SAM-3) for the initiator port associated with every I_T nexus, except the I_T nexus on which the MODE SELECT command was received, with the additional sense code set to LOG PARAMETERS CHANGED.

A TapeAlert Respect Parameter Fields (TARPF) bit set to one specifies that the device server shall select the parameters of the TapeAlert log page (see 8.2.3) to report in response to a LOG SENSE command using the values of the PPC bit and the PARAMETER POINTER field (see SPC-4). A TARPF bit set to zero specifies that the device server shall report all TapeAlert log page parameters regardless of the values of the PPC bit and the PARAMETER POINTER field.

The SHORT ERASE MODE field specifies the action to be taken by the device server when an ERASE (16) or ERASE (6) command with the LONG bit set to 0 is processed. The values for the SHORT ERASE MODE field are specified in table 112. A device server shall support at least one of the specified values.

Table 112 — SHORT ERASE MODE field description

Value	Description
00h	The erase operation shall be performed as specified in SSC-2.
01h	The erase operation shall have no effect on the medium.
02h	The device server shall record an EOD indication at the specified location on the medium.
03h - FFh	Reserved

The Programmable Early Warning Size (PEWS) field specifies the number of megabytes native capacity to use in establishing a PEWZ (see 3.1.55). See 4.2.5 for a description of programmable early warning.

If the volume containing encrypted logical blocks requires encryption (VCELBRE) bit is set set to one and the VCELB bit in the Data Encryption Status page is set to one, then the device server shall require that any logical blocks written to the medium are encrypted (see 4.2.21). If the VCELBRE bit is set to zero then the device server does not use the VCELB bit in the Data Encryption Status page to determine if encryption is required for writing logical blocks.

8.4 Vital product data (VPD) parameters

8.4.1 VPD parameters overview and page codes

This subclause defines the VPD pages used with sequential-access device types. See SPC-4 for VPD pages used with all device types. The VPD page codes specific to sequential-access devices are specified in table 113.

Table 113 — Sequential-access device VPD page codes

Page code	VPD page name	Reference	Support Requirements
B0h	Sequential-access device capabilities VPD page	8.4.2	Optional
B1h	Manufacturer-assigned serial number VPD page	8.4.3	Optional
B2h	TapeAlert supported flags VPD page	8.4.4	Optional
B3h	Automation device serial number VPD page	8.4.5	Optional
B4h – BFh	Reserved for this device type		

8.4.2 Sequential-access device capabilities VPD page

Table 114 specifies the sequential-access device capabilities VPD page. This page provides the application client with the means to determine if the features specified in this page are supported by the device server.

Table 114 — Sequential-access device capabilities VPD page

Bit Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (B0h)							
2	(MSB)							
3	PAGE LENGTH (n-3)							(LSB)
4	Reserved							WORM
n	Reserved							

The PERIPHERAL QUALIFIER field and the PERIPHERAL DEVICE TYPE field are defined in SPC-4.

The PAGE LENGTH field specifies the length of the following VPD page data. If the allocation length value in the INQUIRY command descriptor block is too small to transfer all of the VPD page data, the page length shall not be adjusted to reflect the truncation.

If the Write Once Read Many (WORM) bit is set to one, the device server supports WORM mode operation (see 4.2.20.3). If the WORM bit is set to zero, the device server does not support WORM mode operation.

8.4.3 Manufacturer-assigned serial number VPD page

Table 115 specifies the manufacturer-assigned serial number VPD page.

Table 115 — Manufacturer-assigned serial number VPD page

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (B1h)							
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
4	(MSB)	MANUFACTURER-ASSIGNED SERIAL NUMBER						
n								(LSB)

See SPC-4 for a description of the PERIPHERAL QUALIFIER field, PERIPHERAL DEVICE TYPE field, and PAGE LENGTH field. The PAGE CODE field shall be set to the value specified in table 115.

The MANUFACTURER-ASSIGNED SERIAL NUMBER field contains right-aligned ASCII data (see SPC-4) that is the manufacturer-assigned serial number. If the manufacturer-assigned serial number is not available, the device server shall return ASCII spaces (20h) in this field. If the manufacturer-assigned serial number differs from the value in the PRODUCT SERIAL NUMBER field (see SPC-4) the value in the PRODUCT SERIAL NUMBER field shall be used in building the T10 vendor ID descriptor (see SPC-4).

8.4.4 TapeAlert supported flags VPD page

Table 116 specifies the TapeAlert supported flags VPD page. The TapeAlert supported flags VPD page provides the application client with the means to determine the TapeAlert flags supported by the device server.

Table 116 — TapeAlert supported flags VPD page

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (B2h)							
2	(MSB)	PAGE LENGTH (08h)						
3								(LSB)
4	FLAG01h	FLAG02h	FLAG03h	FLAG04h	FLAG05h	FLAG06h	FLAG07h	FLAG08h
5	FLAG09h	FLAG0Ah	FLAG0Bh	FLAG0Ch	FLAG0Dh	FLAG0Eh	FLAG0Fh	FLAG10h
6	FLAG11h	FLAG12h	FLAG13h	FLAG14h	FLAG15h	FLAG16h	FLAG17h	FLAG18h
7	FLAG19h	FLAG1Ah	FLAG1Bh	FLAG1Ch	FLAG1Dh	FLAG1Eh	FLAG1Fh	FLAG20h
8	FLAG21h	FLAG22h	FLAG23h	FLAG24h	FLAG25h	FLAG26h	FLAG27h	FLAG28h
9	FLAG29h	FLAG2Ah	FLAG2Bh	FLAG2Ch	FLAG2Dh	FLAG2Eh	FLAG2Fh	FLAG30h
10	FLAG31h	FLAG32h	FLAG33h	FLAG34h	FLAG35h	FLAG36h	FLAG37h	FLAG38h
11	FLAG39h	FLAG3Ah	FLAG3Bh	FLAG3Ch	FLAG3Dh	FLAG3Eh	FLAG3Fh	FLAG40h

The PERIPHERAL QUALIFIER field and the PERIPHERAL DEVICE TYPE field are defined in SPC-4.

The PAGE LENGTH field specifies the length of the following VPD page data. If the allocation length value in the INQUIRY command descriptor block is too small to transfer all of the VPD page data, the page length shall not be adjusted to reflect the truncation.

Each FLAGXX bit indicates whether the device server supports the corresponding TapeAlert flag or not. A supported TapeAlert flag has the corresponding FLAGXX bit set to one. A TapeAlert flag that the device server does not support has the corresponding FLAGXX bit set to zero.

8.4.5 Automation device serial number VPD page

Table 117 specifies the automation device serial number VPD page.

Table 117 — Automation device serial number VPD page

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (B3h)							
2	Reserved							
3	PAGE LENGTH (n-3)							
4	(MSB)							
	AUTOMATION DEVICE SERIAL NUMBER							
n								(LSB)

See SPC-4 for a description of the PERIPHERAL QUALIFIER field, PERIPHERAL DEVICE TYPE field, and PAGE LENGTH field. The PAGE CODE field shall be set to the value specified in table 117.

The AUTOMATION DEVICE SERIAL NUMBER field contains the automation device serial number contained in the physical device. If configured (e.g., via the Automation Device Serial Number subpage, see ADC-3), then the automation device serial number shall be the product serial number (see SPC-4) of the media changer containing the physical device under control of the device server (see 4.2.3). If no automation device serial number has been configured, then the device server shall return ASCII spaces (20h) in this field.

8.5 Security protocol parameters

8.5.1 Security protocol overview

This subclause describes the protocols, pages, and descriptors used by sequential-access devices with the SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT commands (see SPC-4).

8.5.2 SECURITY PROTOCOL IN command specifying Tape Data Encryption security protocol

8.5.2.1 SECURITY PROTOCOL IN command specifying Tape Data Encryption security protocol overview

The SECURITY PROTOCOL IN command (see SPC-4) specifying Tape Data Encryption security protocol (i.e., 20h) requests the device server to return information about the data security methods in the device server and on the medium. The command supports a series of pages that are requested individually. An application client

requests a page by using a SECURITY PROTOCOL IN command with the SECURITY PROTOCOL field set to Tape Data Encryption security protocol and the SECURITY PROTOCOL SPECIFIC field set to the page code requested.

A device server that supports the Tape Data Encryption protocol in the SECURITY PROTOCOL OUT command shall also support a SECURITY PROTOCOL IN command specifying the Tape Data Encryption protocol.

The SECURITY PROTOCOL SPECIFIC field (see table 118) specifies the type of report that the application client is requesting.

Table 118 — SECURITY PROTOCOL SPECIFIC field values

Code	Description	Support	Reference
0000h	Tape Data Encryption In Support page	M	8.5.2.2
0001h	Tape Data Encryption Out Support page	M	8.5.2.3
0002h-000Fh	Reserved		
0010h	Data Encryption Capabilities page	E	8.5.2.4
0011h	Supported Key Formats page	E	8.5.2.5
0012h	Data Encryption Management Capabilities page	E	8.5.2.6
0013h-001Fh	Reserved		
0020h	Data Encryption Status page	E	8.5.2.7
0021h	Next Block Encryption Status page	E	8.5.2.8
0022h-002Fh	Reserved		
30h	Random Number page	O	8.5.2.9
31h	Device Server Key Wrapping Public Key page	O	8.5.2.10
0032h-FEFFh	Reserved		
FF00h-FFFFh	Vendor specific		

Support key:
M - mandatory for device servers that support the Tape Data Encryption protocol
E - mandatory for device servers that support the Set Data Encryption page (see 8.5.3.2)
O - optional for device servers that support the Tape Data Encryption protocol

The ALLOCATION LENGTH field specifies the maximum number of bytes that the device server may return (see SPC-4).

8.5.2.2 Tape Data Encryption In Support page

Table 119 specifies the format of the Tape Data Encryption In Support page.

Table 119 — Tape Data Encryption In Support page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1		PAGE CODE (0000h)						(LSB)
2	(MSB)							
3		PAGE LENGTH (n-3)						(LSB)
	Tape Data Encryption In Support page code list							
4	(MSB)							
5		Tape Data Encryption In Support page code (first)						(LSB)
n-1	(MSB)							
n		Tape Data Encryption In Support page code (last)						(LSB)

The Tape Data Encryption In Support page code list shall contain a list of all of the pages that the device server supports for the SECURITY PROTOCOL IN command specifying the Tape Data Encryption security protocol in ascending order beginning with page code 0000h.

8.5.2.3 Tape Data Encryption Out Support page

Table 120 specifies the format of the Tape Data Encryption Out Support page.

Table 120 — Tape Data Encryption Out Support page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1		PAGE CODE (0001h)						(LSB)
2	(MSB)							
3		PAGE LENGTH (n-3)						(LSB)
	Tape Data Encryption Out Support page code list							
4	(MSB)							
5		Tape Data Encryption Out Support page code (first)						(LSB)
n-1	(MSB)							
n		Tape Data Encryption Out Support page code (last)						(LSB)

The Tape Data Encryption Out Support page code list shall contain a list of all of the pages that the device server supports for the SECURITY PROTOCOL OUT command specifying the Tape Data Encryption security protocol in ascending order.

8.5.2.4 Data Encryption Capabilities page

The Data Encryption Capabilities page (see table 121) requests that information regarding the set of data encryption algorithms reported by this device server be sent to the application client. If external data encryption control is supported, then the set of data encryption algorithms reported by the device server may not include all of the algorithms in the set of data encryption algorithms supported by the physical device.

Table 121 — Data Encryption Capabilities page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PAGE CODE (0010h)				(LSB)
2	(MSB)							
3				PAGE LENGTH (n-3)				(LSB)
4	Reserved					EXTDECC		CFG_P
5								
19				Reserved				
	Data Encryption Algorithm descriptor list							
20								
				Data Encryption Algorithm descriptor (first)				
n				Data Encryption Algorithm descriptor (last)				

See SPC-4 for a description of the PAGE CODE field PAGE LENGTH field. The page code field shall be set to the value specified in table 121.

The external data encryption control capable (EXTDECC) field specifies the external data encryption control capability of the physical device. The EXTDECC field values are specified in table 122.

Table 122 — EXTDECC field values

Code	Description
00b	The external data encryption control capability is not supported.
01b	The physical device is not external data encryption control capable.
10b	The physical device is external data encryption control capable.
11b	Reserved

The configuration prevented (CFG_P) field specifies the data encryption parameters configuration capabilities for the algorithms reported in the Data Encryption Algorithm descriptors. The CFG_P field values are specified in table 123.

Table 123 — CFG_P field values

Code	Description
00b	The data encryption configuration capabilities are not reported.
01b	The physical device configured to allow this device server to establish or change data encryption parameters.
10b	The physical device is configured to not allow this device server to establish or change data encryption parameters.
11b	Reserved

Each Data Encryption Algorithm descriptor (see table 124) contains information about a data encryption algorithm supported by the device server. If more than one descriptor is included, they shall be sorted in ascending order of the value in the ALGORITHM INDEX field.

Table 124 — Data Encryption Algorithm descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	ALGORITHM INDEX							
1	Reserved							
2	(MSB)							
3	DESCRIPTOR LENGTH (20)							(LSB)
4	AVFMV	SDK_C	MAC_C	DED_C	DECRYPT_C		ENCRYPT_C	
5	AVFCLP		NONCE_C		Reserved	VCELB_C	UKADF	AKADF
6	(MSB)							
7	MAXIMUM UNAUTHENTICATED KEY-ASSOCIATED DATA BYTES							(LSB)
8	(MSB)							
9	MAXIMUM AUTHENTICATED KEY-ASSOCIATED DATA BYTES							(LSB)
10	(MSB)							
11	KEY SIZE							(LSB)
12	Reserved				RDMC_C			EAREM
13	Reserved							
19								
20	(MSB)							
23	SECURITY ALGORITHM CODE							(LSB)

The ALGORITHM INDEX field is a device server assigned value associated with the algorithm that is being described. The value in the ALGORITHM INDEX field is used by the SECURITY PROTOCOL OUT command Set Data Encryption page to select this algorithm.

The algorithm valid for mounted volume (AVFMV) bit shall be set to one if there is a volume currently mounted and the encryption algorithm being described is valid for that volume. The AVFMV bit shall be set to zero if there is no volume mounted or the algorithm is not valid for the currently mounted volume.

The supplemental decryption key capable (SDK_C) bit shall be set to one if the device server is capable of supporting one or more supplemental decryption keys. The supplemental decryption keys shall be used for decryption only. The SDK_C bit shall be set to zero if the device server is not capable of supporting supplemental decryption keys.

The distinguish encrypted data capable (DED_C) bit shall be set to one if the device server is capable of distinguishing encrypted data from unencrypted data when reading it from the medium. The DED_C bit shall be set to zero if the device server is not capable of distinguishing encrypted data from unencrypted data when reading it from the medium. If the ability to distinguish encrypted data from unencrypted data is format specific and a volume is mounted, the DED_C bit shall be set based on the current format of the medium. If no volume is mounted, the DED_C bit shall be set to one if the device server is capable of distinguishing encrypted data from unencrypted data in any format that the device server supports.

The message authentication code capable (MAC_C) bit shall be set to one if the algorithm includes a message authentication code added to encrypted blocks. The MAC_C bit shall be set to zero if the algorithm does not include a message authentication code added to encrypted blocks. If the inclusion of a message authentication code is format specific and a volume is mounted, the MAC_C bit shall be set based on the current format of the medium. If no volume is mounted, the MAC_C bit shall be set to one if the device server adds a message authentication code to data encrypted with this algorithm in any format that the device server supports.

The DECRYPT_C field (see table 125) specifies the decryption capabilities of the physical device.

Table 125 — DECRYPT_C field values

Code	Name	Description
00b	no capability	The physical device has no has data decryption capability using this algorithm. This value shall be returned if the specified algorithm is disabled.
01b	software capable	The physical device has the ability to decrypt data using this algorithm in software.
10b	hardware capable	The physical device has the ability to decrypt data using this algorithm in hardware.
11b	capable with external control	The physical device has the capability to decrypt data using this algorithm, but control of the data encryption parameters by this device server is prevented.

The ENCRYPT_C field (see table 126) specifies the encryption capabilities of the physical device.

Table 126 — ENCRYPT_C field values

Code	Name	Description
00b	no capability	The physical device has no has data encryption capability using this algorithm. This value shall be returned if the specified algorithm is disabled.
01b	software capable	The physical device has the ability to encrypt data using this algorithm in software.
10b	hardware capable	The physical device has the ability to encrypt data using this algorithm in hardware.
11b	capable with external control	The physical device has the capability to encrypt data using this algorithm, but control of the data encryption parameters by this device server is prevented.

The algorithm valid for current logical position (AVFCLP) field specifies if the encryption algorithm being specified is valid for writing to the mounted volume at the current logical position. Table 127 specifies the values for the AVFCLP field.

Table 127 — AVFCLP field values

Code	Description
00b	Current logical position is not applicable to the encryption algorithm validity or no volume is loaded.
01b	The ecryption algorithm being specified is not valid for writing to the mounted volume at the current logical position.
10b	The ecryption algorithm being specified is valid for writing to the mounted volume at the current logical position.
11b	Reserved

Table 128 specifies the values for the NONCE_C field.

Table 128 — NONCE_C field values

Code	Description
0	This algorithm does not require a nonce value.
1	The device server generates the nonce value.
2	The device server requires all or part of the nonce value to be provided by the application client.
3	The device server supports all or part of the nonce value provided by the application client. If the Set Data Encryption page that enables encryption does not include a nonce value descriptor, the device server generates the nonce value.

If the volume contains encrypted logical block capable (VCELB_C) bit is set to one, then the device server is capable of determining that a volume contains logical blocks encrypted using this algorithm when the volume is mounted. If the VCELB_C is set to zero, then the device server is not capable of determining that a volume contains logical blocks encrypted using this algorithm when the volume is mounted. If the capability of determining that a volume

contains logical blocks encrypted using this algorithm is format specific and a volume is mounted, then the VCELB_C bit is set based on the current format of the medium. If no volume is mounted, the VCELB_C bit is set to one if for at least one algorithm that the device server supports the device server is capable of determining that a volume contains logical blocks encrypted using that algorithm.

The U-KAD Fixed (UKADF) bit shall be set to one if the device server requires the length of U-KAD in the parameter data for a SECURITY PROTOCOL OUT command to equal the value in the MAXIMUM UNAUTHENTICATED KEY-ASSOCIATED BYTES field. If the UKADF bit is set to one, then the MAXIMUM UNAUTHENTICATED KEY-ASSOCIATED BYTES field shall contain a non-zero value. If the UKADF bit is set to zero and the value in the MAXIMUM UNAUTHENTICATED KEY-ASSOCIATED BYTES field is non-zero, then the length of the U-KAD, if present in the parameter data for a SECURITY PROTOCOL OUT command, shall be a value between one and the value in the MAXIMUM UNAUTHENTICATED KEY-ASSOCIATED BYTES field.

The A-KAD Fixed (AKADF) bit shall be set to one if the device server requires the length of A-KAD in the parameter data for a SECURITY PROTOCOL OUT command to equal the value in the MAXIMUM AUTHENTICATED KEY-ASSOCIATED BYTES field. If the AKADF bit is set to one, then the MAXIMUM AUTHENTICATED KEY-ASSOCIATED BYTES field shall contain a non-zero value. If the AKADF bit is set to zero and the value in the MAXIMUM AUTHENTICATED KEY-ASSOCIATED BYTES field is non-zero, then the length of the A-KAD, if present in the parameter data for a SECURITY PROTOCOL OUT command, shall be a value between one and the value in the MAXIMUM AUTHENTICATED KEY-ASSOCIATED BYTES field.

The MAXIMUM UNAUTHENTICATED KEY-ASSOCIATED DATA BYTES field indicates the maximum size of the unauthenticated key-associated data (see 4.2.21.13) that the device server can support for this algorithm.

The MAXIMUM AUTHENTICATED KEY-ASSOCIATED DATA BYTES field indicates the maximum size of the authenticated key-associated data (see 4.2.21.13) that the device server can support for this algorithm.

The KEY SIZE field indicates the size in bytes of the encryption key required by the algorithm.

The raw decryption mode control capabilities (RDMC_C) field indicates the capabilities the encryption algorithm provides to the application client to control read operations that access encrypted blocks while the decryption mode is set to RAW. Table 129 defines the values for the RDMC_C field.

Table 129 — RDMC_C field values

Code	Description
0h	No capabilities are specified.
1h	The encryption algorithm does not allow read operations in RAW decryption mode.
2h-3h	Reserved
4h	The encryption algorithm disables read operations in RAW mode by default and allows the application client to control RAW reads via the RDMC field in the Set Data Encryption page (see 8.5.3.2).
5h	The encryption algorithm enables read operations in RAW mode by default and allows the application client to control RAW reads via the RDMC field in the Set Data Encryption page (see 8.5.3.2).
6h	The encryption algorithm disables read operations in RAW mode by default and does not allow the application client to control RAW reads via the RDMC field in the Set Data Encryption page (see 8.5.3.2).
7h	The encryption algorithm enables read operations in RAW mode by default and does not allow the application client to control RAW reads via the RDMC field in the Set Data Encryption page (see 8.5.3.2).

The encryption algorithm records encryption mode (EAREM) bit shall be set to one if the encryption mode is recorded with each encrypted block. The EAREM bit shall be set to zero if the encryption mode is not recorded with each encrypted block.

The SECURITY ALGORITHM CODE field contains an security algorithm code (see SPC-4).

8.5.2.5 Supported Key Formats page

Table 130 specifies the format of the Supported Key Formats page.

Table 130 — Supported Key Formats page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			PAGE CODE (0011h)				
1								(LSB)
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
4				SUPPORTED KEY FORMATS LIST				
n								

The SUPPORTED KEY FORMATS LIST field shall contain a list of all of the key formats (see table 147) that the device server supports for the Set Data Encryption page (see 8.5.3.2) in ascending order.

8.5.2.6 Data Encryption Management Capabilities page

Table 131 specifies the format of the Data Encryption Management Capabilities page.

Table 131 — Data Encryption Management Capabilities page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PAGE CODE (0012h)				(LSB)
2	(MSB)							
3				PAGE LENGTH (12)				(LSB)
4	Reserved							LOCK_C
5	Reserved					CKOD_C	CKORP_C	CKORL_C
6	Reserved							
7	Reserved					AITN_C	LOCAL_C	PUBLIC_C
8				Reserved				
15								

The LOCK_C bit shall be set to one if the device server supports the LOCK bit in the Set Data Encryption page. The LOCK_C bit shall be set to zero if the device server does not support the LOCK bit in the Set Data Encryption page.

The CKOD_C bit shall be set to one if the device server supports the CKOD bit in the Set Data Encryption page. The CKOD_C bit shall be set to zero if the device server does not support the CKOD bit in the Set Data Encryption page.

The CKORP_C bit shall be set to one if the device server supports the CKORP bit in the Set Data Encryption page. The CKORP_C bit shall be set to zero if the device server does not support the CKORP bit in the Set Data Encryption page.

The CKORL_C bit shall be set to one if the device server supports the CKORL bit in the Set Data Encryption page. The CKORL_C bit shall be set to zero if the device server does not support the CKORL bit in the Set Data Encryption page.

The AITN_C bit shall be set to one if the device server supports a scope of ALL I_T NEXUS. The AITN_C bit shall be set to zero if the device server does not support a scope of ALL I_T NEXUS.

The LOCAL_C bit shall be set to one if the device server supports a scope of LOCAL. The LOCAL_C bit shall be set to zero if the device server does not support a scope of LOCAL.

The PUBLIC_C bit shall be set to one if the device server supports a scope of PUBLIC. The PUBLIC_C bit shall be set to zero if the device server does not support a scope of PUBLIC.

8.5.2.7 Data Encryption Status page

Table 132 specifies the format of the Data Encryption Status page.

Table 132 — Data Encryption Status page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PAGE CODE (0020h)				(LSB)
2	(MSB)							
3				PAGE LENGTH (n-3)				(LSB)
4	I_T NEXUS SCOPE			Reserved		KEY SCOPE		
5	ENCRYPTION MODE							
6	DECRYPTION MODE							
7	ALGORITHM INDEX							
8	(MSB)							
11				KEY INSTANCE COUNTER				(LSB)
12	Reserved	PARAMETERS CONTROL			VCELB	CEEMS		RDMD
13								
23				Reserved				
24								
n				KEY-ASSOCIATED DATA DESCRIPTORS LIST				

The I_T NEXUS SCOPE field shall contain the value from the data encryption scope saved for the I_T nexus on which this command was received (see 4.2.21.7).

The KEY SCOPE field shall contain the value from the key scope in the saved data encryption parameters currently associated with the I_T nexus on which this command was received (see 4.2.21.8).

The ENCRYPTION MODE field shall contain the value from the encryption mode in the saved data encryption parameters currently associated with the I_T nexus on which this command was received (see 4.2.21.8).

The DECRYPTION MODE field shall contain the value from the decryption mode in the saved data encryption parameters currently associated with the I_T nexus on which this command was received (see 4.2.21.8).

The ALGORITHM INDEX field shall contain the value from the algorithm index in the saved data encryption parameters currently associated with the I_T nexus on which this command was received (see 4.2.21.8). If the ENCRYPTION MODE field and the DECRYPTION MODE field are both set to DISABLE, the value in the ALGORITHM INDEX field is undefined.

The KEY INSTANCE COUNTER field contains the value of the key instance counter (see 4.2.21.10) assigned to the key indicated by the KEY SCOPE field value.

The PARAMETERS CONTROL field specifies information on how the data encryption parameters are controlled. The PARAMETERS CONTROL field values are specified in table 133.

Table 133 — PARAMETERS CONTROL field values

Code	Description
000b	Data encryption parameters control is not reported.
001b	Data encryption parameters are not exclusively controlled by external data encryption control.
010b	Data encryption parameters are exclusively controlled by the sequential-access device server.
011b	Data encryption parameters are not exclusively controlled by the automation/drive interface device server.
100b	Data encryption parameters are not exclusively controlled by a management interface.
101b-111b	Reserved

If the VCELB_C bit is set to one in the Data Encryption Capabilities page, then the volume contains encrypted logical blocks (VCELB) bit shall be set to one when a mounted volume contains an encrypted logical block. The VCELB bit shall be set to zero if:

a) the mounted volume does not contain any encrypted logical blocks;
b) there is no volume mounted; or
c) the VCELB_C bit in the Data Encryption Capabilities page is set to zero.

The raw decryption mode disabled (RDMD) bit shall be set to one if the device server is configured to mark each encrypted record as disabled for raw read operations based on the RDMC_C value and the raw decryption mode disable parameter in the saved data encryption parameters currently associated with the I_T nexus on which the command was received (see 4.2.21.7).

The check external encryption mode status (CEEMS) field shall contain the value from the check external encryption mode parameter in the saved data encryption parameters currently associated with the I_T nexus on which the command was received (see 4.2.21.7).

If the ENCRYPTION MODE field and the DECRYPTION MODE field are both set to DISABLE, the KEY-ASSOCIATED DATA DESCRIPTORS LIST field shall not be included in the page.

If either the ENCRYPTION MODE field or the DECRYPTION MODE field is set to a value other than DISABLE, the KEY-ASSOCIATED DATA DESCRIPTORS LIST field shall contain data security descriptors (see 8.5) describing attributes assigned to the key defined by the I_T NEXUS SCOPE and KEY SCOPE fields at the time the key was established in the device server. If more than one key associated descriptor is included, they shall be in order of increasing value of the DESCRIPTOR TYPE field. Descriptors shall be included as defined by the following paragraphs.

An unauthenticated key-associated data descriptor (see 8.5.4.3) shall be included if an unauthenticated key-associated data descriptor was included when the key was established in the device server. The AUTHENTICATED field is reserved. The KEY DESCRIPTOR field shall contain the U-KAD value associated with the key.

An authenticated key-associated data descriptor (see 8.5.4.4) shall be included if an authenticated key-associated data descriptor was included when the key was established in the device server. The AUTHENTICATED field is reserved. The KEY DESCRIPTOR field shall contain the A-KAD value associated with the key.

A nonce value descriptor (see 8.5.4.5) shall be included if a nonce value descriptor was included when the key was established in the device server. The AUTHENTICATED field is reserved. The KEY DESCRIPTOR field shall contain the

nonce value associated with the key. A nonce value descriptor may be included if no nonce value descriptor was included when the key was established in the device server. In this case, the KEY DESCRIPTOR field shall be set to the nonce value established by the device server for use with the selected key.

A metadata key-associated data descriptor (see 8.5.4.6) shall be included if the metadata key-associated data descriptor was included when the data encryption parameters were established. The KEY DESCRIPTOR field shall contain the M-KAD value associated with the key.

8.5.2.8 Next Block Encryption Status page

Table 134 specifies the format of the Next Block Encryption Status page.

Table 134 — Next Block Encryption Status page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PAGE CODE (0021h)				(LSB)
2	(MSB)							
3				PAGE LENGTH (n-3)				(LSB)
4	(MSB)							
11				LOGICAL OBJECT NUMBER				(LSB)
12	COMPRESSION STATUS				ENCRYPTION STATUS			
13	ALGORITHM INDEX							
14	Reserved						EMES	RDMDS
15	Reserved							
16								
n				KEY-ASSOCIATED DATA DESCRIPTORS				

The LOGICAL OBJECT NUMBER field contains the logical object identifier of the next logical object (see 4.2.7.2).

The COMPRESSION STATUS field values are specified in table 135.

Table 135 — COMPRESSION STATUS field values

Code	Description
0h	The device server is incapable of determining if the logical object referenced by the LOGICAL OBJECT NUMBER field has been compressed.
1h	The device server is capable of determining if the logical object referenced by the LOGICAL OBJECT NUMBER field has been compressed, but is not able to at this time. Possible reasons are: a) the next logical block has not yet been read into the buffer; b) there was an error reading the next logical block; or c) there are no more logical blocks (i.e., end-of-data).
2h	The device server has determined that the logical object referenced by the LOGICAL OBJECT NUMBER field is not a logical block.
3h	The device server has determined that the logical object referenced by the LOGICAL OBJECT NUMBER field is not compressed.
4h	The device server has determined that the logical object referenced by the LOGICAL OBJECT NUMBER field is compressed.
5h-Fh	Reserved

The ENCRYPTION STATUS field values are specified in table 136.

Table 136 — ENCRYPTION STATUS field values

Code	Description
0h	The device server is incapable of determining if the logical object referenced by the LOGICAL OBJECT NUMBER field has been encrypted.
1h	The device server is capable of determining if the logical object referenced by the LOGICAL OBJECT NUMBER field has been encrypted, but is not able to at this time. Possible reasons are: a) the next logical block has not yet been read into the buffer; b) there was an error reading the next logical block; or c) there are no more logical blocks (i.e., end-of-data).
2h	The device server has determined that the logical object referenced by the LOGICAL OBJECT NUMBER field is not a logical block.
3h	The device server has determined that the logical object referenced by the LOGICAL OBJECT NUMBER field is not encrypted.
4h	The device server has determined that the logical object referenced by the LOGICAL OBJECT NUMBER field is encrypted by an algorithm that is not supported by this device server. The values in the KEY-ASSOCIATED DATA DESCRIPTORS field contain information pertaining to the encrypted block.
5h	The device server has determined that the logical object referenced by the LOGICAL OBJECT NUMBER field is encrypted by an algorithm that is supported by this device server. The values in the ALGORITHM INDEX and KEY-ASSOCIATED DATA DESCRIPTORS fields contain information pertaining to the encrypted block.
6h	The device server has determined that the logical object referenced by the LOGICAL OBJECT NUMBER field is encrypted by an algorithm that is supported by this device server, but the device server is either not enabled to decrypt or does not have the correct key or nonce value to decrypt the encrypted block.
7h-Fh	Reserved

The ALGORITHM INDEX field indicates which of the encryption algorithms reported by the SECURITY PROTOCOL IN command Data Encryption Capabilities page was used to encrypt the logical block. For values in the ENCRYPTION STATUS field (see table 136) that do not indicate the ALGORITHM INDEX field is valid, the algorithm index is undefined.

The encryption mode external status (EMES) bit shall be set to one if:

a) the ENCRYPTION STATUS field is set to either 5h or 6h;
b) the EAREM bit in the algorithm descriptor (see 8.5.2.4) for the algorithm specified by the ALGORITHM INDEX field is set to one; and
c) the next block is marked as having been written to the medium while the encryption mode was set to EXTERNAL.

The EMES bit shall be set to zero if:

a) the ENCRYPTION STATUS field is set to a value other than 5h or 6h;
b) the EAREM bit in the algorithm descriptor (see 8.5.2.4) for the algorithm specified by the ALGORITHM INDEX field is set to zero; or
c) the next block is marked as having been written to the medium while the encryption mode was set to ENCRYPT.

The raw decryption mode disabled status (RDMDS) bit shall be set to one if the device server supports raw decryption mode, the ENCRYPTION STATUS field is set to either 5h or 6h, and the next block is marked as disabled for raw decryption mode operations (see 4.2.21). The RDMDS bit shall be set to zero if the device server does not support raw decryption mode, the ENCRYPTION STATUS field is set to a value other than 5h or 6h, or the next block is not marked as disabled for raw decryption mode operations.

If the encryption status indicates that the next logical block is encrypted by a supported algorithm, the device server shall include in the KEY-ASSOCIATED DATA DESCRIPTORS field all key-associated data that is associated with the encrypted block that was recorded on the medium. If more than one key-associated data descriptor is include in the Next Block Encryption Status page, they shall be in increasing numeric order of the value in the DESCRIPTOR TYPE field.

An unauthenticated key-associated data descriptor (see 8.5.4.3) shall be included if any unauthenticated key-associated data is associated with the next logical block. The AUTHENTICATED field shall be set to 1. The KEY DESCRIPTOR field shall contain the U-KAD value associated with the encrypted block.

An authenticated key-associated data descriptor (see 8.5.4.4) shall be included if any authenticated key-associated data is associated with the next logical block. The AUTHENTICATED field shall indicate the status of the authentication done by the device server (see table 157). The KEY DESCRIPTOR field shall contain the A-KAD value associated with the encrypted block.

The Next Block Encryption Status page may include a nonce value descriptor (see 8.5.4.5). If a nonce value descriptor is included, then the AUTHENTICATED field shall indicate the status of the authentication done by the device server (see table 157). The KEY DESCRIPTOR field shall contain the nonce value associated with the encrypted block.

A metadata key-associated data descriptor (see 8.5.4.6) shall be included if any M-KAD is associated with the next logical block and the decryption mode is set to RAW in the saved data encryption parameters currently associated with the I_T nexus on which this command was received. The KEY DESCRIPTOR field shall contain the M-KAD value associated with the encrypted block.

8.5.2.9 Random Number page

Table 137 specifies the format of the Random Number page.

Table 137 — Random Number page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PAGE CODE (0030h)				(LSB)
2	(MSB)							
3				PAGE LENGTH (32)				(LSB)
4	(MSB)							
35				RANDOM NUMBER				(LSB)

The RANDOM NUMBER field contains a secure random number (see SPC-4), suitable for use as a random nonce (see SPC-4) that is generated by the device server using a source of entropy available within the device. Each request for the Random Number page shall generate a new secure random number for the RANDOM NUMBER field.

8.5.2.10 Device Server Key Wrapping Public Key page

8.5.2.10.1 Device Server Key Wrapping Public Key page overview

The Device Server Key Wrapping Public Key page may be used by an application client to read the device server's key wrapping public key. The format of the Device Server Key Wrapping Public Key page is specified in table 138.

Table 138 — Device Service Key Wrapping Public Key page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PAGE CODE (0031h)				(LSB)
2	(MSB)							
3				PAGE LENGTH (n-3)				(LSB)
4	(MSB)							
7				PUBLIC KEY TYPE				(LSB)
8	(MSB)							
11				PUBLIC KEY FORMAT				(LSB)
12	(MSB)							
13				PUBLIC KEY LENGTH (n-9)				(LSB)
14	(MSB)							
n				PUBLIC KEY				(LSB)

The PUBLIC KEY TYPE field specifies the type of public key in the PUBLIC KEY field. The PUBLIC KEY TYPE field values are specified in table 139.

Table 139 — PUBLIC KEY TYPE field values

Code	Description	Reference
00000000h	RSA 2048	8.5.2.10.2
00000001h-0000000Fh	Reserved	
00000010h	ECC 521	8.5.2.10.3
00000011h-FFFFBFFFh	Reserved	
FFFFC000h-FFFFFFFFh	Vendor specific	

The PUBLIC KEY FORMAT, PUBLIC KEY LENGTH, and PUBLIC KEY field values depend on the PUBLIC KEY TYPE field.

8.5.2.10.2 RSA 2048 public keys

The PUBLIC KEY FORMAT field shall be set to 00000000h. All other values for the PUBLIC KEY FORMAT field are reserved. The PUBLIC KEY LENGTH field shall be set to 512. Bytes 14 through 269 shall contain the modulus n. Bytes 270 through 525 shall contain the public exponent e (see PKCS #1 V2.1).

8.5.2.10.3 ECC 521 public keys

The PUBLIC KEY FORMAT field shall be set to 00000000h. All other values for the PUBLIC KEY FORMAT field are reserved. The PUBLIC KEY LENGTH field shall be set to 133. Bytes 14 through 146 shall contain the ECC 521 public key as converted by the algorithm specified in ANSI X9.63 section 4.3.6 using the uncompressed form.

8.5.3 SECURITY PROTOCOL OUT command specifying Tape Data Encryption security protocol

8.5.3.1 SECURITY PROTOCOL OUT command specifying Tape Data Encryption security protocol overview

The SECURITY PROTOCOL OUT command specifying the Tape Data Encryption security protocol (i.e., 20h) is used to configure the data security methods in the device server and on the medium. The command supports a series of pages that are sent individually. An application client requests to send a page by using a SECURITY PROTOCOL OUT command with the SECURITY PROTOCOL field set to Tape Data Encryption security protocol and the SECURITY PROTOCOL SPECIFIC field set to the page code requested.

The SECURITY PROTOCOL SPECIFIC field (see table 140) specifies the type of page that the application client is sending.

Table 140 — SECURITY PROTOCOL SPECIFIC field values

Code	Description	Reference
0000h-000Fh	Reserved	
0010h	Set Data Encryption page	8.5.3.2
0011h	SA Encapsulation page	8.5.3.3
0012h-002Fh	Reserved	
0030h-003Fh	Restricted	ADC-3
0040h-FEFFh	Reserved	
FF00h-FFFFh	Vendor specific	

If the SECURITY PROTOCOL SPECIFIC field is set to a reserved or unsupported value, the device server shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

8.5.3.2 Set Data Encryption page

8.5.3.2.1 Set Data Encryption page overview

Table 141 specifies the format of the Set Data Encryption page.

Table 141 — Set Data Encryption page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1	PAGE CODE (0010h)							(LSB)
2	(MSB)							
3	PAGE LENGTH (m-3)							(LSB)
4	SCOPE			Reserved				LOCK
5	CEEM		RDMC		SDK	CKOD	CKORP	CKORL
6	ENCRYPTION MODE							
7	DECRYPTION MODE							
8	ALGORITHM INDEX							
9	KEY FORMAT							
10	Reserved							
17								
18	(MSB)							
19	KEY LENGTH (n-19)							(LSB)
20	KEY							
n								
n+1	KEY-ASSOCIATED DATA DESCRIPTORS LIST							
m								

The PAGE LENGTH field specifies the number of bytes of parameter data to follow. If the page length value results in the truncation of any field, the device server shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The SCOPE field (see table 142) specifies the scope of the data encryption parameters. Support for scope values of PUBLIC and ALL I_T NEXUS are mandatory for device servers that support the Set Data Encryption page.

Table 142 — SCOPE field values

Code	Name	Description
0	PUBLIC	All fields other than the scope field and LOCK bit shall be ignored. The I_T nexus shall use data encryption parameters that are shared by other I_T nexuses. If no I_T nexuses are sharing data encryption parameters, the device server shall use default data encryption parameters.
1	LOCAL	The data encryption parameters are unique to the I_T nexus associated with the SECURITY PROTOCOL OUT command and shall not be shared with other I_T nexuses.
2	ALL I_T NEXUS	The data encryption parameters shall be shared with all I_T nexuses.
3-7		Reserved

See 4.2.21.11 for a description of the LOCK bit.

Table 143 describes the values for the check external encryption mode (CEEM) field..

Table 143 — CEEM field values

Code	Description
00b	Vendor specific
01b	Do not check the encryption mode that was in use when the block was written to the medium.
10b	On read and verify commands, check the encryption mode that was in use when the block was written to the medium. Report an error if the block was written in EXTERNAL mode (see 4.2.21.5).
11b	On read and verify commands, check the encryption mode that was in use when the block was written to the medium. Report an error if the block was written in ENCRYPT mode (see 4.2.21.5).

The device server shall terminate the SECURITY PROTOCOL OUT command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER DATA if the CEEM field is set to either 10b or 11b, and:

a) the DECRYPTION MODE field is set to DISABLE; or
b) the EAREM bit in the algorithm descriptor (see 8.5.2.4) for the algorithm specified by the ALGORITHM INDEX field is set to zero.

The raw decryption mode control (RDMC) field specifies if the device server shall mark each encrypted block written to the medium as disabled for read operations in raw mode (i.e., read operations with the decryption mode set to

RAW). The RDMC field shall be ignored if the ENCRYPTION MODE field is not set to ENCRYPT. Table 144 specifies the values for the RDMC field when the ENCRYPTION MODE field is set to ENCRYPT.

Table 144 — RDMC field values

Code	Description
00b	The device server shall mark each encrypted block per the default setting for the algorithm (see table 129).
01b	Reserved
10b	The device server shall mark each encrypted block written to the medium in a format specific manner as enabled for raw decryption mode operations.
11b	The device server shall mark each encrypted block written to the medium in a format specific manner as disabled for raw decryption mode operations.

The device server shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST and the additional sense key set to INVALID FIELD IN PARAMETER DATA if:

a) the ENCRYPTION MODE field is set to ENCRYPT;
b) the RDMC field is set to 10b or 11b; and
c) the RDMC_C field in the algorithm descriptor for the encryption algorithm selected by the value in the ALGORITHM INDEX field is set to 1h, 6h, or 7h.

If the supplemental decryption key (SDK) bit is set to one, the key sent in this page shall be added to the set of data encryption parameters used by the device server for the selected scope. The KEY INSTANCE COUNTER shall not be incremented for supplemental decryption keys. The ENCRYPTION MODE and LOCK fields shall be ignored and the DECRYPTION MODE shall match the current setting for this scope. If the DECRYPTION MODE does not match the current settings for this scope, the device server shall terminate the command with CHECK CONDITION status and set the sense key to ILLEGAL REQUEST and the additional sense code to INVALID FIELD IN PARAMETER LIST.

If the physical device does not currently have a saved set of data encryption parameters associated with the I_T nexus that sent the Set Data Encryption page or the scope or decryption mode values do not match the values in that set of saved data encryption parameters, the device server shall terminate the command with CHECK CONDITION status and set the sense key to ILLEGAL REQUEST and the additional sense code to INVALID FIELD IN PARAMETER LIST.

If the SDK bit is set to one and the SDK_C field is set to zero in the Data Encryption Algorithm descriptor field that matches the ALGORITHM INDEX in the Data Encryption capabilities page, the device server shall terminate the command with CHECK CONDITION status and set the sense key set ILLEGAL REQUEST and the additional sense code to INVALID FIELD IN PARAMETER LIST.

If the device server is processing a Set Data Encryption page with the SDK bit set to one and does not have the resource available to store this key the device server shall terminate the command with CHECK CONDITION status and set the sense key to ILLEGAL REQUEST and the additional sense code to MAXIMUM NUMBER OF SUPPLEMENTAL DECRYPTION KEYS EXCEEDED. Any previously saved supplemental decryption keys shall not be affected by this error.

If the SDK bit is set to zero, the key sent in this page shall be the key used for both encryption and decryption. Any keys that have been previously stored by the device server shall be removed from memory. See 4.2.21.6.

If the clear key on demount (CKOD) bit is set to one the physical device shall set the data encryption parameters to default values upon completion of a volume demount. If the CKOD bit is set to zero, the demounting of a volume

shall not affect the data encryption parameters. If the CKOD bit is set to one and there is no volume mounted, the device server shall terminate the command with CHECK CONDITION status and set the sense key to ILLEGAL REQUEST and the additional sense code to INVALID FIELD IN PARAMETER DATA.

If the clear key on reservation preempt (CKORP) bit is set to one the physical device shall set the data encryption parameters to default values when a persistent reservation is preempted (i.e., a PERSISTENT RESERVE OUT command specifying a service action of PREEMPT or PREEMPT AND ABORT is processed). If the CKORP bit is set to zero, a preemption of a persistent reservation shall not affect the data encryption parameters. If the CKORP bit is set to one and there is no persistent reservation in effect for the I_T nexus associated with the SECURITY PROTOCOL OUT command, the device server shall terminate the command with CHECK CONDITION status and set the sense key to ILLEGAL REQUEST and the additional sense code to INVALID FIELD IN PARAMETER DATA.

If the clear key on reservation loss (CKORL) bit is set to one the physical device shall set the data encryption parameters to default values on a reservation loss (see 3.1.56). If the CKORL bit is set to zero, a reservation loss shall not affect the data encryption parameters. If the CKORL bit is set to one and there is no reservation in effect for the I_T nexus associated with the SECURITY PROTOCOL OUT command, the device server shall terminate the command with CHECK CONDITION status and set the sense key to ILLEGAL REQUEST and the additional sense code to INVALID FIELD IN PARAMETER DATA.

Table 145 specifies the values for the ENCRYPTION MODE field.

Table 145 — ENCRYPTION MODE field values

Code	Name	Description
0h	DISABLE	Data encryption is disabled.
1h	EXTERNAL	The data associated with the WRITE(6) and WRITE(16) commands has been encrypted by a system that is compatible with the algorithm specified by the ALGORITHM INDEX field.
2h	ENCRYPT	The device server shall encrypt all data that it receives for a WRITE(6) or WRITE(16) command using the algorithm specified in the ALGORITHM INDEX field and the key specified in the KEY field.
3h-Fh		Reserved

Table 146 specifies the values for the DECRYPTION MODE field. See 4.2.21.3 for configuration and exception condition requirements.

Table 146 — DECRYPTION MODE field values

Code	Name	Description
0h	DISABLE	Data decryption is disabled. If the device server encounters an encrypted logical block while reading, it shall not allow access to the data.
1h	RAW	Data decryption is disabled. If the device server encounters an encrypted logical block while reading, it shall pass the encrypted block to the host without decrypting it. The encrypted block may contain data that is not user data.
2h	DECRYPT	The device server shall decrypt all data that is read from the medium when processing a READ(6), READ(16), READ REVERSE(6), READ REVERSE(16), or RECOVER BUFFERED DATA command or verified when processing a VERIFY(6) or VERIFY(16) command. The data shall be decrypted using the algorithm specified in the ALGORITHM INDEX field and the key specified in the KEY field.
3h	MIXED	The device server shall decrypt all data that is read from the medium that the device server determines was encrypted when processing a READ(6), READ(16), READ REVERSE(6), READ REVERSE(16), or RECOVER BUFFERED DATA command or verified when processing a VERIFY(6) or VERIFY(16) command. The data shall be decrypted using the algorithm specified in the ALGORITHM INDEX field and the key specified in the KEY field. If the device server encounters unencrypted data when processing a READ(6), READ(16), READ REVERSE(6), READ REVERSE(16), RECOVER BUFFERED DATA, VERIFY(6), or VERIFY(16) command, the data shall be processed without decrypting.
4h-Fh		Reserved

If the physical device is not capable of distinguishing encrypted blocks from unencrypted blocks using the algorithm specified in the ALGORITHM INDEX field and the DECRYPTION MODE field is set to MIXED, the device server shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER DATA.

If the ENCRYPTION MODE field is set to ENCRYPT and the KEY LENGTH field is set to zero, the device server shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER DATA.

If the DECRYPTION MODE field is set to DECRYPT or MIXED and the KEY LENGTH field is set to zero, the device server shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER DATA.

The ALGORITHM INDEX field indicates which of the encryption algorithms reported by the SECURITY PROTOCOL IN command Data Encryption Capabilities pages shall be used to encrypt and decrypt data. If the algorithm specified in the ALGORITHM INDEX field is disabled, then the deevice server shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to ENCRYPTION ALGORITHM DISABLED.

If a volume is mounted and the combination of the ENCRYPTION MODE, DECRYPTION MODE, and ALGORITHM INDEX fields is not valid for the mounted volume or current logical position, the device server shall terminate the command

with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the addition sense code set to INVALID FIELD IN PARAMETER DATA.

The KEY FORMAT field specifies the format of the value in the KEY field. Values for the KEY FORMAT field are specified in table 147.

Table 147 — KEY FORMAT field values

Code	Description	Reference
00h	The KEY field contains the key to be used to encrypt or decrypt data.	8.5.3.2.2
01h	The KEY field contains a vendor-specific key reference.	8.5.3.2.3
02h	The KEY field contains the key wrapped by the device server public key.	8.5.3.2.4
03h	The KEY field contains a key that is encrypted using ESP-SCSI.	8.5.3.2.5
04h-BFh	Reserved	
C0h-FFh	Vendor specific	

Editors Note 4 - DAP: Would like to use the term logical block instead of data throughout the security text. Make the change but keep in mind it may be better to use encrypted block in some instances.

The KEY LENGTH field specifies the length of the KEY field in bytes.

If the ENCRYPTION MODE field is set to ENCRYPT the device server shall save the key-associated descriptors in the KEY-ASSOCIATED DATA DESCRIPTORS LIST field and associate them with every logical block that is encrypted with this key by the device server.

If the ENCRYPTION MODE field is set to EXTERNAL the device server shall save the key-associated descriptors in the KEY-ASSOCIATED DATA DESCRIPTORS LIST field and associate them with every logical block that is written using the data encryption parameters established by this command.

If more than one key-associated data descriptor is specified in the Set Data Encryption page, they shall be in increasing numeric order of the value in the DESCRIPTOR TYPE field.

The device server shall terminate the command with CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST if:

a) key-associated descriptors are included in the KEY-ASSOCIATED DATA DESCRIPTORS LIST field;
b) DECRYPTION MODE field is not set to RAW; and;
c) the ENCRYPTION MODE field is not set to:
   A) EXTERNAL; or
   B) ENCRYPT.

An unauthenticated key-associated data descriptor (see 8.5.4.3) may be included if any unauthenticated key-associated data is to be associated with logical blocks encrypted with the algorithm and key. The AUTHENTICATED field is reserved. The KEY DESCRIPTOR field shall contain the U-KAD value associated with the encrypted block. The device server shall terminate the command with CHECK CONDITION status, with the sense key set to

ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER DATA if the UKADF bit is set to one in the data encryption algorithm descriptor, the ENCRYPTION MODE field is set to ENCRYPT, and:

a) the length of the KEY DESCRIPTOR field is not equal to the value in the MAXIMUM UNAUTHENTICATED KEY-ASSOCIATED BYTES field of the data encryption algorithm descriptor; or
b) the parameter data does not contain an unauthenticated key-associated data descriptor.

An authenticated key-associated data descriptor (see 8.5.4.4) may be included if any authenticated key-associated data is to be associated with logical blocks encrypted with the algorithm and key. The AUTHENTICATED field is reserved. The KEY DESCRIPTOR field shall contain the A-KAD value associated with the encrypted block. The device server shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER DATA if the AKADF bit is set to one in the data encryption algorithm descriptor, the ENCRYPTION MODE field is set to ENCRYPT, and:

a) the length of the KEY DESCRIPTOR field is not equal to the value in the MAXIMUM AUTHENTICATED KEY-ASSOCIATED BYTES field of the data encryption algorithm descriptor; or
b) the parameter data does not contain an authenticated key-associated data descriptor.

If a nonce value descriptor (see 8.5.4.5) is included and the algorithm and the device server supports application client generated nonce values, the value in the KEY DESCRIPTOR field shall be used as the nonce value for the encryption process. If a nonce value descriptor is included and the encryption algorithm or the device server does not support application client generated nonce values, the device server shall terminate the command with CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST. If the encryption algorithm or the device server requires an application client generated nonce value and a nonce value descriptor is not included, the device server shall terminate the command with CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INCOMPLETE KEY-ASSOCIATED DATA SET. If a nonce value descriptor is included, the AUTHENTICATED field is reserved. The KEY DESCRIPTOR field shall contain the nonce value associated with the encrypted block.

A metadata key-associated data descriptor (see 8.5.4.6) may be included if the DECRYPTION MODE field is set to RAW and the encryption algorithm requires any metadata key-associated data to be associated with encrypted logical blocks read when the DECRYPTION MODE field is set to RAW.

A metadata key-associated data descriptor (see 8.5.4.6) shall be included if the ENCRYPTION MODE field is set to EXTERNAL and the encryption algorithm requires any metadata key-associated data to be associated with logical blocks written when the ENCRYPTION MODE field is set to EXTERNAL.

The device server shall terminate the command with CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST if an M-KAD is included and:

a) the ENCRYPTION MODE field is not set to EXTERNAL and the DECRYPTION MODE field is not set to RAW; or,
b) the encryption algorithm specified by the ALGORITHM INDEX field does not support M-KAD.

8.5.3.2.2 Plain-text key

If the KEY FORMAT field is set to 00h, the KEY field contains the key in an algorithm-specific format. Table 148 specifies the format of the key in the KEY field if the KEY FORMAT field is set to 00h.

Table 148 — KEY field format with KEY FORMAT field set to 00h

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			KEY				
n								(LSB)

8.5.3.2.3 Key reference

If the KEY FORMAT field is set to 01h, the KEY field shall contain 8 bytes of T10 vendor identification (see SPC-4) followed by a vendor-specific key reference identifying the key to be used to encrypt or decrypt data. If the KEY field contains a vendor-specific key reference that is unknown to the device server, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to VENDOR SPECIFIC KEY REFERENCE NOT FOUND. Table 149 specifies the format of the KEY field if the KEY FORMAT field is set to 01h.

Table 149 — KEY field format with KEY FORMAT field set to 01h

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			T10 VENDOR IDENTIFICATION				
7								(LSB)
8	(MSB)			VENDOR-SPECIFIC KEY REFERENCE				
n								(LSB)

8.5.3.2.4 Key wrapped by device server public key

8.5.3.2.4.1 Key wrapped by device server public key overview

Table 150 specifies the format of the KEY field if the KEY FORMAT field is set to 02h.

Table 150 — KEY field format with KEY FORMAT field set to 02h

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			PARAMETER SET				
1								(LSB)
2	(MSB)			LABEL LENGTH				
3								(LSB)
4	(MSB)			LABEL				
n								(LSB)
n+1	(MSB)			WRAPPED KEY LENGTH				
n+2								(LSB)
n+3	(MSB)			WRAPPED KEY				
m								(LSB)
m+1	(MSB)			SIGNATURE LENGTH				
m+2								(LSB)
m+3	(MSB)			SIGNATURE				
z								(LSB)

The PARAMETER SET field specifies the parameters used in the key wrapping operation. The PARAMETER SET field values are specified in table 151.

Table 151 — PARAMETER SET field values

Code	Description	Reference
0000h	RSA 2048	8.5.3.2.4.2
0001h-000Fh	Reserved	
0010h	ECC 521	8.5.3.2.4.3
0011h-BFFFh	Reserved	
C000h-FFFFh	Vendor specific	

The LABEL LENGTH field specifies the length of the LABEL field in bytes.

The LABEL field contains public information associated with the data encryption key (e.g., key identification). The LABEL field shall consist of Wrapped Key descriptors (see 8.5.4.7).

The WRAPPED KEY LENGTH field specifies the length of the WRAPPED KEY field in bytes.

The WRAPPED KEY field contains the data encryption key encrypted by the parameters specified in the PARAMETER SET field.

The SIGNATURE LENGTH field specifies the length of the SIGNATURE field in bytes.

The SIGNATURE field contains the wrapper's signature as specified by the PARAMETER SET field.

If the device server determines that an incorrect public key was used to wrap the data encryption key, the device server shall terminate the command with CHECK CONDITION status, set the sense key to DATA PROTECT, and set the additional sense code to INCORRECT DATA ENCRYPTION KEY.

If a device server that tries to verify the signature determines that it does not have the specified signature verification public key, the device server shall terminate the command with CHECK CONDITION status, set the sense key to DATA PROTECT, and set the additional sense code to UNKNOWN SIGNATURE VERIFICATION KEY.

If the device server encounters an error while unwrapping the data encryption key, the device server shall terminate the command with CHECK CONDITION status, set the sense key to DATA PROTECT, and set the additional sense code to UNABLE TO DECRYPT DATA.

If a device server encounters an error while verifying the signature over the wrapped data encryption key, the device server shall terminate the command with CHECK CONDITION status, set the sense key to DATA PROTECT, and set the additional sense code to CRYPTOGRAPHIC INTEGRITY VALIDATION FAILED.

8.5.3.2.4.2 Key wrapping with RSA 2048

If the PARAMETER SET field is set to 0000h, the WRAPPED KEY field shall contain the data encryption key wrapped using RSAES-OAEP as specified by PKCS #1 V2.1. The RSA modulus length shall be 2048 bits. The hash function used shall be SHA-256. The mask generation function (MGF) used shall be MGF1 with SHA-256 as the hash function.

The WRAPPED KEY shall be the data encryption key encrypted with the device server's public key as specified by PKCS #1 V2.1 in 7.1.1 RSAES-OAEP-ENCRYPT ((n, e), M, L), where (n, e) is the public key, M is the raw key, and L is the LABEL field as specified in 8.5.3.2.4.1.

If a signature is included, the SIGNATURE field shall be the RSASSA-PSS signature over the WRAPPED KEY field as specified by PKCS #1 V2.1 in 8.1.1 using the wrapping entity's private key. The RSAES-OAEP-ENCRYPT operation adds an integrity check to both L (i.e., the label) and M (i.e., the key). The signature is for providing proof of the wrapper's identification. The device server may check the signature to verify that the key was wrapped and signed by a valid key management server.

8.5.3.2.4.3 Key wrapping with ECC 521

If the PARAMETER SET field is set to 0010h, the WRAPPED KEY field shall contain the data encryption key wrapped using ECIES-HC as specified in ISO/IEC 18033-2. The ECIES-HC requirements and parameters for ECIES-KEM are specified in table 152.

Table 152 — ECIES-HC requirements and parameters for ECIES-KEM

Item	Parameter	Description	Reference
Elliptic curve	Elliptic curve	Curve P-521	Appendix 6 FIPS 186-2 Change 1
KDF		Approved Alternative 1	NIST SP800-56A
	H	SHA-512	
	AlgorithmID	00001h	
	CofactorMode	0	
	OldCofactor-Mode	0	
	CheckMode	1	
	SingleHash-Mode	0	
	KeyLen	96	
	fmt	uncompressed	

NOTE 72 The deviation of the definition of the KDF from ISO/IEC 18033-2 is to allow a conforming implementation to be FIPS 140-2 compliant.

The ECIES-HC requirements and parameters for ECIES-DEM are specified in table 153.

Table 153 — ECIES-HC requirements and parameters for ECIES-DEM

Item	Parameter	Description
DEM		DEM1
	SC	SC1 using AES-256 as the block cipher
	MA	HMAC with SHA-512

If a signature is included, the SIGNATURE field shall be the ECDSA signature over the WRAPPED KEY field as specified in FIPS 186-2 using the wrapping entity's private key. The device server may check the signature to verify that the key was wrapped and signed by a valid key management server.

8.5.3.2.5 Key encrypted using ESP-SCSI

If the KEY FORMAT field is set to 03h, then the KEY field shall contain an ESP-SCSI out w/o length descriptor (see SPC-4) that includes a key that has been encrypted in accordance with an SA that has been created in the device server (see SPC-4). The SA shall use an encryption algorithm other than ENCR_NULL. The KEY LENGTH field contains the length of the ESP-SCSI out w/o length descriptor.

If the USAGE_TYPE SA parameter in the SA associated with the value in the DS_SAI field in the ESP-SCSI out w/o length descriptor is not set to 0081h (i.e. Tape Data Encryption), then the device server shall terminate the

command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID SA USAGE.

8.5.3.3 SA Encapsulation page

The SA Encapsulation page shall contain an ESP-SCSI out descriptor (see SPC-4) that has been encrypted in accordance with an SA that has been created in the device server (see SPC-4). The SA shall use an encryption algorithm other than ENCR_NULL.

If the USAGE_TYPE SA parameter in the SA associated with the value in the DS_SAI field in the ESP-SCSI out descriptor is not set to 0081h (i.e., Tape Data Encryption), then the device server shall terminate the command with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID SA USAGE.

The format of the SA Encapsulation page is specified in table 154.

Table 154 — SA Encapsulation page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PAGE CODE (0011h)				(LSB)
2								
n				ESP-SCSI out descriptor				

See SPC-4 for a description of the ESP-SCSI out descriptor. The ENCRYPTED OR AUTHENTICATED DATA field in the ESP-SCSI out descriptor contains any Tape Data Encryption security protocol SECURITY PROTOCOL OUT command page except the SA Encapsulation page.

8.5.4 SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT descriptors

8.5.4.1 Tape Data Encryption security protocol descriptors overview

Several of the parameter pages in used by the SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT commands allow for the inclusion of descriptors to provide additional optional data. This subclause defines the descriptors that are common between multiple pages.

8.5.4.2 Tape Data Encryption descriptors format

Each Tape Data Encryption descriptor shall be of the format as specified in table 155.

Table 155 — Tape Data Encryption descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	KEY DESCRIPTOR TYPE							
1	Reserved						AUTHENTICATED	
2	(MSB)							
3	KEY DESCRIPTOR LENGTH (n-3)							(LSB)
4	(MSB)							
n	KEY DESCRIPTOR							(LSB)

The KEY DESCRIPTOR TYPE field contains a value from table 156 that defines the contents of the KEY DESCRIPTOR field.

Table 156 — KEY DESCRIPTOR TYPE field values

Code	Description	Reference
00h	Unauthenticated key-associated data	8.5.4.3
01h	Authenticated key-associated data	8.5.4.4
02h	Nonce value	8.5.4.5
03h	Metadata key-associated data	8.5.4.6
04-BFh	Reserved	
C0h-FFh	Vendor specific	

Table 157 defines the values for the AUTHENTICATED field.

Table 157 — AUTHENTICATED field values

Code	Description
0	Reserved
1	The value in the KEY DESCRIPTOR field is not covered by the authentication (i.e., U-KAD).
2	No attempt has been made to authenticate the value in the KEY DESCRIPTOR field.
3	The value in the KEY DESCRIPTOR field has been authenticated.
4	The value in the KEY DESCRIPTOR field has failed authenticated.
5-7	Reserved

8.5.4.3 Unauthenticated key-associated data key descriptor

The AUTHENTICATED field in an unauthenticated key-associated data descriptor shall be set to 1.

The KEY DESCRIPTOR field of an unauthenticated key-associated data descriptor shall contain any unauthenticated key-associated data assigned to the key.

8.5.4.4 Authenticated key-associated data key descriptor

The AUTHENTICATED field shall be set as defined by the page in which the descriptor is included.

The KEY DESCRIPTOR field of an authenticated key-associated data descriptor shall contain any authenticated key-associated data assigned to the key.

8.5.4.5 Nonce value descriptor

The AUTHENTICATED field shall be set as defined by the page in which the descriptor is included.

The KEY DESCRIPTOR field of a nonce value descriptor shall contain the nonce value used with the data encryption key.

8.5.4.6 Metadata key-associated data key descriptor

The AUTHENTICATED field in an M-KAD descriptor shall be set to 2h.

The KEY DESCRIPTOR field of a M-KAD descriptor contains data required by the encryption algorithm for a keyless copy operation (see 4.2.21.5).

8.5.4.7 Wrapped Key descriptors

The format of a Wrapped Key descriptor is specified in table 158.

Table 158 — Wrapped Key descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	WRAPPED KEY DESCRIPTOR TYPE							
1	Reserved							
2	(MSB)							
3	WRAPPED KEY DESCRIPTOR LENGTH (n-3)							(LSB)
4	(MSB)							
n	WRAPPED KEY DESCRIPTOR							(LSB)

The WRAPPED KEY DESCRIPTOR TYPE field contains a value from table 159 that defines the contents of the WRAPPED KEY DESCRIPTOR field.

Table 159 — WRAPPED KEY DESCRIPTOR TYPE field values

Code	Description	Reference	Support
00h	Device server identification descriptor	8.5.4.8	M
01h	Key wrapping entity identification descriptor	8.5.4.9	M
02h	Wrapped key information descriptor	8.5.4.10	O
03h	Wrapped key identification descriptor	8.5.4.11	M
04h	Wrapped key length descriptor	8.5.4.12	M
05h-BFh	Reserved		
C0h-FFh	Vendor-specific descriptor		

Support key:
M - mandatory for device servers that support the Device Server Key Wrapping Public Key page (see 8.5.2.10).
O - optional for device servers that support the Device Server Key Wrapping Public Key page (see 8.5.2.10).

8.5.4.8 Device server identification descriptor

The WRAPPED KEY DESCRIPTOR field shall contain the logical unit name (see SAM-4) for the device server. The logical unit name may be used by the device server to validate that the data encryption key was wrapped with the correct public key. The logical unit name may also be used by an application client to validate that the data encryption key was wrapped with the correct public key, if the key was wrapped by a third-party system (e.g., a key management server)

8.5.4.9 Key wrapping entity identification descriptor

The WRAPPED KEY DESCRIPTOR field shall contain identification data that uniquely identifies the key wrapping entity. The identification data may be used by the device server to locate the key wrapper's public key for signature verification.

8.5.4.10 Wrapped key information descriptor

The WRAPPED KEY DESCRIPTOR field contains information that may be used to identify the key. This information is not required to be unique (e.g., a label entered by the system operator).

8.5.4.11 Wrapped key identification descriptor

The WRAPPED KEY DESCRIPTOR field shall contain the key identifer.

8.5.4.12 Wrapped key length descriptor

The WRAPPED KEY DESCRIPTOR field shall contain the length of the wrapped key in bytes.

Annex A
(informative)

A.1 Application client recommendations for using TapeAlert

A.1.1 Overview

The previous version of this standard included some recommendations for application clients that use TapeAlert. This standard deprecates these recommendations as they fall outside of its scope. To minimize the effort of referencing these recommendations for those application client designers that have incorporated them into products, this standard gathers them together and presents them below.

A.1.2 Recommendations for using TapeAlert

Upon detecting an active TapeAlert flag, the application client should:

a) communicate an error message, the flag's severity (see table 10), and the applicable description for that severity (see table 9) to the user interface; and
b) store the error message, the flag's severity, and the applicable attributes for that severity in a log.

At the beginning of each set of TapeAlert error messages, the application client should identify the SCSI target device that initiated them. For a medium-related flag (e.g., flags 04h, 07h, and 0Fh), the application client should include the software label of the medium in the TapeAlert error message so that the user is aware what piece of media the error refers to. Such information may be displayed with the messages for other flags.

Each time the application client reads the TapeAlert log page or the TapeAlert Response log page, it should check all returned flags (see table 10) to detect active flags. More than one flag may be active at a time.

The information read in the TapeAlert flags should not, in itself, cause the application client to stop a data transfer sequence (e.g., a backup or restore operation).

A.2 TapeAlert flag associated information

Table A.1 lists the TapeAlert flags and associated information.

Table A.1 — TapeAlert log page parameter codes

Code	Flag	Recommended application client message	Probable cause
01h	Read warning	The tape drive is having problems reading data. No data has been lost, but there has been a reduction in the performance of the tape.	The drive is having severe trouble reading.
02h	Write warning	The tape drive is having problems writing data. No data has been lost, but there has been a reduction in the capacity of the tape.	The drive is having severe trouble writing.
a) Media Recognition System (MRS) is a method where pre-defined stripes are placed at the beginning of the media to identify the media. The MRS stripes are read to determine if the media is of data-grade. Data-grade media should be used in SCSI streaming devices since it is of the required quality and consistency to be used to store data (i.e., audio/video grade media should not be used).			

Table A.1 — TapeAlert log page parameter codes (Continued)

Code	Flag	Recommended application client message	Probable cause
03h	Hard error	The operation has stopped because an error has occurred while reading or writing data that the drive cannot correct.	The drive had a hard read or write error.
04h	Media	Your data is at risk: 1. Copy any data you require from this tape. 2. Do not use this tape again. 3. Restart the operation with a different tape.	Media can no longer be written/read, or performance is severely degraded.
05h	Read failure	The tape is damaged or the drive is faulty. Call the tape drive supplier help line.	The drive can no longer read data from the tape.
06h	Write failure	The tape is from a faulty batch or the tape drive is faulty: 1. Use a good tape to test the drive. 2. If the problem persists, call the tape drive supplier help line.	The drive can no longer write data to the tape.
07h	Media life	The tape cartridge has reached the end of its calculated useful life: 1. Copy any data you need to another tape. 2. Discard the old tape.	The media has exceeded its specified life.
08h	Not data grade	The cartridge is not data-grade. Any data you write to the tape is at risk. Replace the cartridge with a data-grade tape.	The drive has not been able to read the MRS[a] stripes.
09h	Write protect	You are trying to write to a write protected cartridge. Remove the write protection or use another tape.	Write command is attempted to a write protected tape.
0Ah	No removal	You cannot eject the cartridge because the tape drive is in use. Wait until the operation is complete before ejecting the cartridge.	Manual or software unload attempted when prevent media removal is on.
0Bh	Cleaning media	The tape in the drive is a cleaning cartridge.	Cleaning tape loaded into drive.
0Ch	Unsupported format	You have tried to load a cartridge of a type that is not supported by this drive.	Attempted load of unsupported tape format (e.g., DDS2 in DDS1 drive).
0Dh	Recoverable mechanical cartridge failure	The operation has failed because the tape in the drive has experienced a mechanical failure: 1. Discard the old tape. 2. Restart the operation with a different tape.	Tape snapped/cut or other cartridge mechanical failure in the drive where medium can be demounted.

a) Media Recognition System (MRS) is a method where pre-defined stripes are placed at the beginning of the media to identify the media. The MRS stripes are read to determine if the media is of data-grade. Data-grade media should be used in SCSI streaming devices since it is of the required quality and consistency to be used to store data (i.e., audio/video grade media should not be used).

Table A.1 — TapeAlert log page parameter codes (Continued)

Code	Flag	Recommended application client message	Probable cause
0Eh	Unrecoverable mechanical cartridge failure	The operation has failed because the tape in the drive has experienced a mechanical failure: 1. Do not attempt to extract the tape cartridge. 2. Call the tape drive supplier help line.	Tape snapped/cut or other cartridge mechanical failure in the drive where medium cannot be demounted.
0Fh	Memory chip in cartridge failure	The memory in the tape cartridge has failed, which reduces performance. Do not use the cartridge for further write operations.	Memory chip failed in cartridge.
10h	Forced eject	The operation has failed because the tape cartridge was manually demounted while the tape drive was actively writing or reading.	Manual or forced eject while drive actively writing or reading.
11h	Read only format	You have loaded a cartridge of a type that is read-only in this drive. The cartridge will appear as write protected.	Media loaded that is read-only format.
12h	Tape directory corrupted on load	The tape directory on the tape cartridge has been corrupted. File search performance will be degraded. The tape directory can be rebuilt by reading all the data on the cartridge.	Tape drive powered down with tape loaded, or permanent error prevented the tape directory being updated.
13h	Nearing media life	The tape cartridge is nearing the end of its calculated life. It is recommended that you: 1. Use another tape cartridge for your next backup. 2. Store this tape cartridge in a safe place in case you need to restore data from it.	Media may have exceeded its specified number of passes.
14h	Cleaning required	The tape drive needs cleaning: 1. If the operation has stopped, eject the tape and clean the drive. 2. If the operation has not stopped, wait for it to finish and then clean the drive. Check the tape drive users manual for device specific cleaning instructions.	The drive thinks it has a head clog or needs cleaning.
15h	Cleaning requested	The tape drive is due for routine cleaning: 1. Wait for the current operation to finish. 2. Then use a cleaning cartridge. Check the tape drive users manual for device specific cleaning instructions.	The drive is ready for a periodic cleaning.
16h	Expired cleaning media	The last cleaning cartridge used in the tape drive has worn out: 1. Discard the worn out cleaning cartridge. 2. Wait for the current operation to finish. 3. Then use a new cleaning cartridge.	The cleaning tape has expired.

a) Media Recognition System (MRS) is a method where pre-defined stripes are placed at the beginning of the media to identify the media. The MRS stripes are read to determine if the media is of data-grade. Data-grade media should be used in SCSI streaming devices since it is of the required quality and consistency to be used to store data (i.e., audio/video grade media should not be used).

Table A.1 — TapeAlert log page parameter codes (Continued)

Code	Flag	Recommended application client message	Probable cause
17h	Invalid cleaning tape	The last cleaning cartridge used in the tape drive was an invalid type: 1. Do not use this cleaning cartridge in this drive. 2. Wait for the current operation to finish. 3. Then use a valid cleaning cartridge.	Invalid cleaning tape type used.
18h	Retension requested	The tape drive has requested a retension operation.	The drive is having severe trouble reading or writing, that will be resolved by a retension cycle.
19h	Dual-port interface error	A redundant interface port on the tape drive has failed.	Failure of one interface port in a dual-port configuration (i.e., Fibre Channel)
1Ah	Cooling fan failure	A tape drive cooling fan has failed.	Fan failure inside tape drive mechanism or tape drive enclosure.
1Bh	Power supply failure	A redundant power supply has failed inside the tape drive enclosure. Check the enclosure users manual for instructions on replacing the failed power supply.	Redundant PSU failure inside the tape drive enclosure or rack subsystem.
1Ch	Power consumption	The tape drive power consumption is outside the specified range.	Power consumption of the tape drive is outside specified range.
1Dh	Drive maintenance	Preventive maintenance of the tape drive is required. Check the tape drive users manual for device specific preventive maintenance tasks or call the tape drive supplier help line.	The drive requires preventive maintenance (not cleaning).
1Eh	Hardware A	The tape drive has a hardware fault: 1. Eject the tape or magazine. 2. Reset the drive. 3. Restart the operation.	The drive has a hardware fault that requires reset to recover.
1Fh	Hardware B	The tape drive has a hardware fault: 1. Turn the tape drive off and then on again. 2. Restart the operation. 3. If the problem persists, call the tape drive supplier help line.	The drive has a hardware fault that is not read/write related or requires a power cycle to recover.
20h	Interface	The tape drive has a problem with the application client interface: 1. Check the cables and cable connections. 2. Restart the operation.	The drive has identified an interface fault.

a) Media Recognition System (MRS) is a method where pre-defined stripes are placed at the beginning of the media to identify the media. The MRS stripes are read to determine if the media is of data-grade. Data-grade media should be used in SCSI streaming devices since it is of the required quality and consistency to be used to store data (i.e., audio/video grade media should not be used).

Table A.1 — TapeAlert log page parameter codes (Continued)

Code	Flag	Recommended application client message	Probable cause
21h	Eject media	The operation has failed: 1. Eject the tape or magazine. 2. Insert the tape or magazine again. 3. Restart the operation.	Error recovery action.
22h	Microcode update fail	The microcode update has failed because you have tried to use the incorrect microcode for this tape drive. Obtain the correct microcode and try again.	Microcode update failed.
23h	Drive humidity	Environmental conditions inside the tape drive are outside the specified humidity range.	Drive humidity limits exceeded.
24h	Drive temperature	Environmental conditions inside the tape drive are outside the specified temperature range.	Cooling problem.
25h	Drive voltage	The voltage supply to the tape drive is outside the specified range.	Drive voltage limits exceeded.
26h	Predictive failure	A hardware failure of the tape drive is predicted. Call the tape drive supplier help line.	Predictive failure of drive hardware.
27h	Diagnostics required	The tape drive may have a hardware fault. Run extended diagnostics to verify and diagnose the problem. Check the tape drive users manual for device specific instructions on running extended diagnostic tests.	The drive may have a hardware fault that may be identified by extended diagnostics (i.e., SEND DIAGNOSTIC command).
28h - 2Eh	Obsolete		
2Fh - 31h	Rsvd		
32h	Lost statistics	Media statistics have been lost at some time in the past.	Drive or library powered on with tape loaded.
33h	Tape directory invalid at unload	The tape directory on the tape cartridge just unloaded has been corrupted. File search performance will be degraded. The tape directory can be rebuilt by reading all the data.	Error preventing the tape directory being updated on unload.
34h	Tape system area write failure	The tape just unloaded could not write its system area successfully: 1. Copy data to another tape cartridge. 2. Discard the old cartridge.	Write errors while writing the system area on unload.
35h	Tape system area read failure	The tape system area could not be read successfully at load time: 1. Copy data to another tape cartridge.	Read errors while reading the system area on load.

a) Media Recognition System (MRS) is a method where pre-defined stripes are placed at the beginning of the media to identify the media. The MRS stripes are read to determine if the media is of data-grade. Data-grade media should be used in SCSI streaming devices since it is of the required quality and consistency to be used to store data (i.e., audio/video grade media should not be used).

Table A.1 — TapeAlert log page parameter codes (Continued)

Code	Flag	Recommended application client message	Probable cause
36h	No start of data	The start of data could not be found on the tape: 1. Check that you are using the correct format tape. 2. Discard the tape or return the tape to your supplier.	Tape damaged, bulk erased, or incorrect format.
37h	Loading failure	The operation has failed because the media cannot be loaded and threaded. 1. Remove the cartridge, inspect it as specified in the product manual, and retry the operation. 2. If the problem persists, call the tape drive supplier help line.	The drive is unable to load the media and thread the tape.
38h	Unrecoverable unload failure	The operation has failed because the medium cannot be unloaded: 1. Do not attempt to extract the tape cartridge. 2. Call the tape driver supplier help line.	The drive is unable to unload the medium.
39h	Automation interface failure	The tape drive has a problem with the automation interface: 1. Check the power to the automation system. 2. Check the cables and cable connections. 3. Call the supplier help line if problem persists.	The drive has identified an interface fault.
3Ah	Microcode failure	The tape drive has reset itself due to a detected microcode fault. If problem persists, call the supplier help line.	Microcode bug.
3Bh	WORM Medium - Integrity Check Failed	The tape drive has detected an inconsistency during the WORM medium integrity checks. Someone may have tampered with the cartridge.	Someone has tampered with the WORM medium.
3Ch	WORM Medium - Overwrite Attempted	An attempt had been made to overwrite user data on a WORM medium: 1. If a WORM medium was used inadvertently, replace it with a normal data medium. 2. If a WORM medium was used intentionally: a) check that the software application is compatible with the WORM medium format you are using. b) check that the medium is bar-coded correctly for WORM.	The application software does not recognize the medium as WORM.
3Dh	Rsvd		
3Eh	Rsvd		
3Fh	Rsvd		
40h	Rsvd		

a) Media Recognition System (MRS) is a method where pre-defined stripes are placed at the beginning of the media to identify the media. The MRS stripes are read to determine if the media is of data-grade. Data-grade media should be used in SCSI streaming devices since it is of the required quality and consistency to be used to store data (i.e., audio/video grade media should not be used).

Annex B
(informative)

B.1 Security environment

B.1.1 Security environment overview

Figure B.1 shows a simplified depiction of a typical security deployment, in which a large number of device servers are being accessed by a set of data management servers, each set controlled by a master data management server. The addition of encryption in the device servers introduces a centralized key manager that may be part of the master data management server. Centralized key managers are used for the same reason that they use master data management servers, so that relatively few key managers are installed in an environment with a relatively large number of device servers and data management servers.

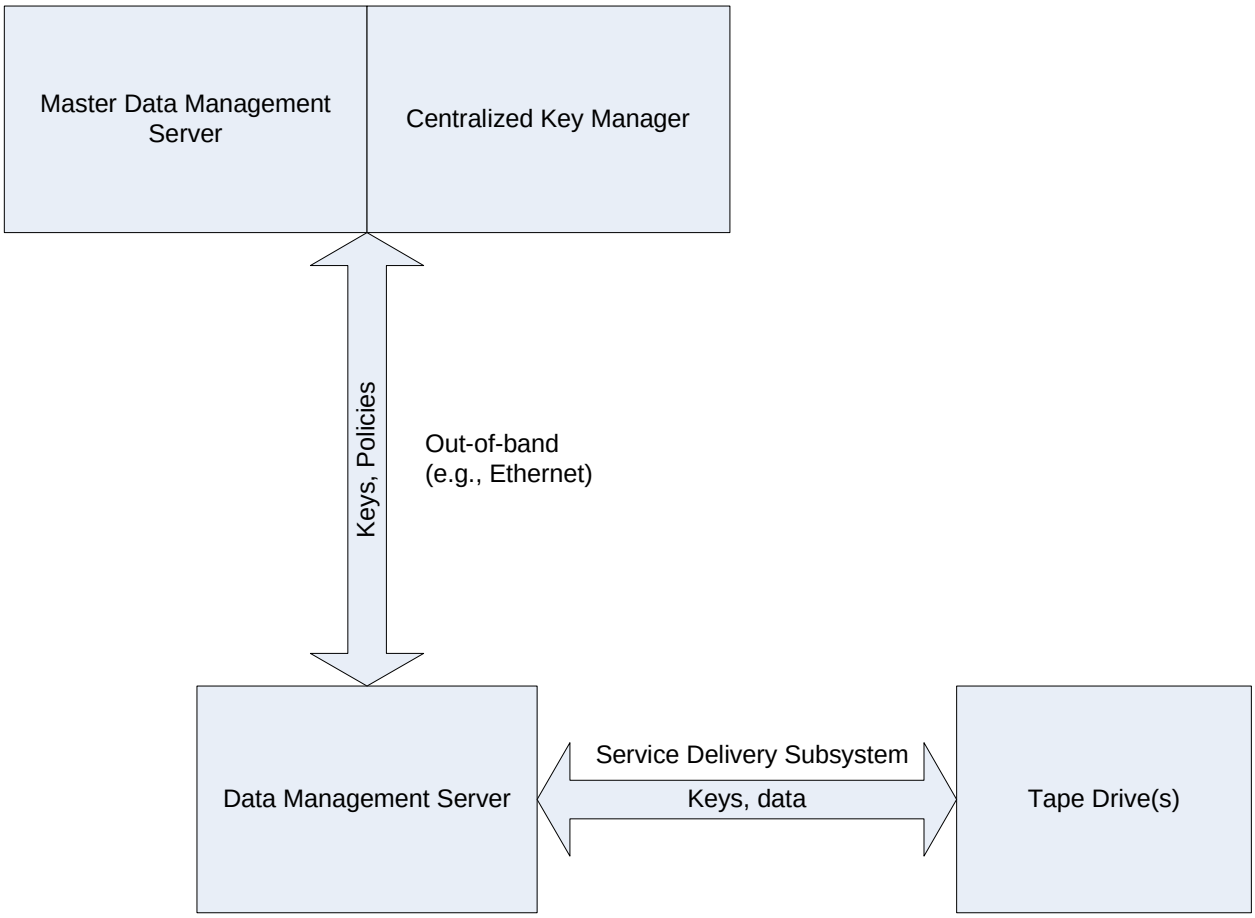

Figure B.1 — Simple security deployment environment

Communication between the key manager, master server, and the data management server is beyond the scope of this standard, while communication between the data management server and device server is over a SCSI service delivery subsystem. The following additional assumptions are in place:

a) the communications path between the key manager and the data management server can easily be made sufficiently secure (e.g. via SSL with a 128 bit strong cipher suite is probably sufficient for most link based attacks);
b) the key manager is able to reliably associate keys with data management servers;
c) only the data management server is able to associate keys with data attributes;
d) the device server has a minimal cryptographic module, able to encrypt and decrypt data, and to perform simple key load/key agreement operations;
e) attacks against key disclosure or substitution for keys already stored within the device server, or Key Manager are out of scope; and
f) the data management server, in most cases, has only a simple SSL module implemented in the underlying OS.

As a result of the above, the assumption is threats are against either the data management server or the service delivery subsystem.

B.1.2 Security environment threats

Table B.1 specifies a list of security environment threats and methods to mitigate the threats.

Table B.1 — Security environment threats

Threat	Against data management server	Against service delivery subsystem
Subverting key load protocol via CDB modification (e.g. turning off data encryption).	Harden the OS of the data management server.	Secure the channel from the data management server to the device server.
Key disclosure (passive).	a) encrypt keys prior to reaching the data management server; or b) prevent information leakage by hardening the OS of the data management server to prevent caching or core dumps, etc.	a) only send encrypted keys along service delivery subsystem network, or b) secure the channel from the data management server to the device server.
Injecting an attacker's key.	a) encrypt and sign keys at the key manager prior to shipment to the data management server; or b) harden the OS of the data management server.	a) only send signed keys along the T10 network; or b) secure the channel from the data management server to the device server.
Breaking association of keys to data via key replay.	Harden the OS of the data management server.	a) rotate signing keys frequently; or b) have a secure channel from the data management server to the device server.

Annex C
(informative)

C.1 Flowchart of example keyless copy operation

Figure C.1 shows a simplified flowchart of an example keyless copy operation.

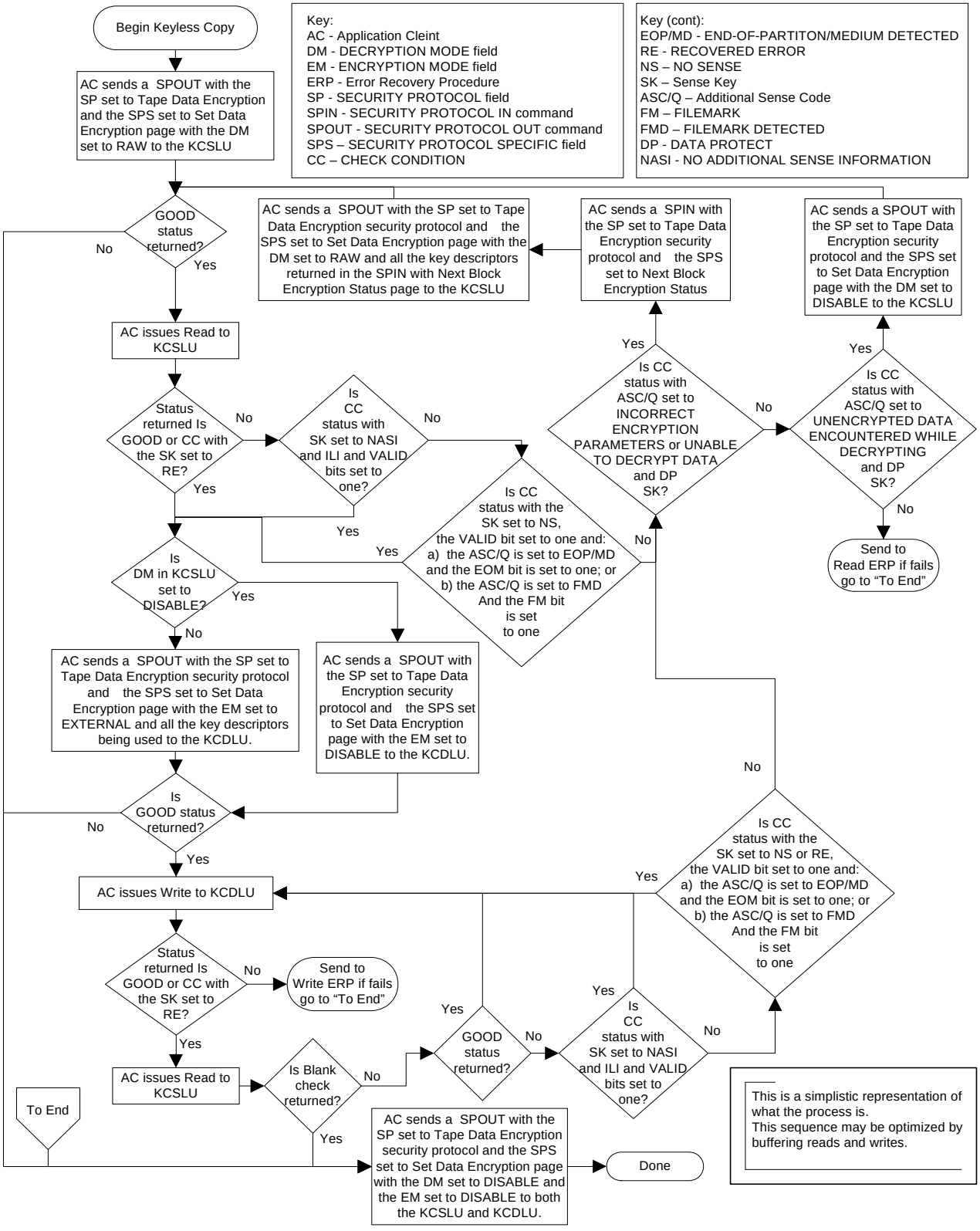

Figure C.1 — Example keyless copy operation flowchart